



UvA-DARE (Digital Academic Repository)

H-AMR

A New GPU-accelerated GRMHD Code for Exascale Computing with 3D Adaptive Mesh Refinement and Local Adaptive Time Stepping

Liska, M.T.P.; Chatterjee, K.; Issa, D.; Yoon, D.; Kaaz, N.; Tchekhovskoy, A.; van Eijnatten, D.; Musoke, G.; Hesp, C.; Rohoza, V.; Markoff, S.; Ingram, A.; van der Klis, M.

DOI

[10.3847/1538-4365/ac9966](https://doi.org/10.3847/1538-4365/ac9966)

Publication date

2022

Document Version

Final published version

Published in

Astrophysical Journal, Supplement Series

License

CC BY

[Link to publication](#)

Citation for published version (APA):

Liska, M. T. P., Chatterjee, K., Issa, D., Yoon, D., Kaaz, N., Tchekhovskoy, A., van Eijnatten, D., Musoke, G., Hesp, C., Rohoza, V., Markoff, S., Ingram, A., & van der Klis, M. (2022). H-AMR: A New GPU-accelerated GRMHD Code for Exascale Computing with 3D Adaptive Mesh Refinement and Local Adaptive Time Stepping. *Astrophysical Journal, Supplement Series*, 263(2), [26]. <https://doi.org/10.3847/1538-4365/ac9966>

General rights

It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations

If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: <https://uba.uva.nl/en/contact>, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (<https://dare.uva.nl>)



H-AMR: A New GPU-accelerated GRMHD Code for Exascale Computing with 3D Adaptive Mesh Refinement and Local Adaptive Time Stepping

M. T. P. Liska¹, K. Chatterjee¹, D. Issa², D. Yoon³, N. Kaaz² , A. Tchekhovskoy², D. van Eijnatten³, G. Musoke³, C. Hesp³, V. Rohoza², S. Markoff³ , A. Ingram⁴ , and M. van der Klis³

¹ Institute for Theory and Computation, Harvard University, 60 Garden Street, Cambridge, MA 02138, USA; matthew.liska@cfa.harvard.edu

² Center for Interdisciplinary Exploration & Research in Astrophysics (CIERA), Physics & Astronomy, Northwestern University, Evanston, IL 60202, USA

³ Anton Pannekoek Institute for Astronomy (API), University of Amsterdam, Science Park 904, Amsterdam, The Netherlands

⁴ School of Mathematics, Statistics and Physics, Newcastle University, Herschel Building, Newcastle upon Tyne NE1 7RU, UK

Received 2022 July 7; revised 2022 October 7; accepted 2022 October 10; published 2022 November 22

Abstract

General relativistic magnetohydrodynamic (GRMHD) simulations have revolutionized our understanding of black hole accretion. Here, we present a GPU-accelerated GRMHD code H-AMR with multifaceted optimizations that, collectively, accelerate computation by 2–5 orders of magnitude for a wide range of applications. First, it introduces a spherical grid with 3D adaptive mesh refinement that operates in each of the three dimensions independently. This allows us to circumvent the Courant condition near the polar singularity, which otherwise cripples high-resolution computational performance. Second, we demonstrate that local adaptive time stepping on a logarithmic spherical-polar grid accelerates computation by a factor of $\lesssim 10$ compared to traditional hierarchical time-stepping approaches. Jointly, these unique features lead to an effective speed of $\sim 10^9$ zone cycles per second per node on 5400 NVIDIA V100 GPUs (i.e., 900 nodes of the OLCF Summit supercomputer). We illustrate H-AMR's computational performance by presenting the first GRMHD simulation of a tilted thin accretion disk threaded by a toroidal magnetic field around a rapidly spinning black hole. With an effective resolution of $13,440 \times 4608 \times 8092$ cells and a total of $\lesssim 22$ billion cells and $\sim 0.65 \times 10^8$ time steps, it is among the largest astrophysical simulations ever performed. We find that frame dragging by the black hole tears up the disk into two independently precessing subdisks. The innermost subdisk rotation axis intermittently aligns with the black hole spin, demonstrating for the first time that such long-sought alignment is possible in the absence of large-scale poloidal magnetic fields.

Unified Astronomy Thesaurus concepts: [Black hole physics \(159\)](#); [High energy astrophysics \(739\)](#); [Magnetohydrodynamical simulations \(1966\)](#)

Supporting material: [animation](#)

1. Introduction

GRMHD simulations in conjunction with radiative transfer calculations provide arguably the most direct link between the physical laws describing the motion of gas and magnetic fields near black holes and the observed phenomenology of accretion disks and jets in astrophysical black hole systems. Initially, the numerical work focused on the geometrically thick, radiatively inefficient accretion flows (RIAFs, e.g., Narayan & Yi 1994, 1995), with the accretion typically proceeding in the equatorial plane of a spinning black hole (e.g., De Villiers et al. 2003; Hirose et al. 2004; McKinney & Gammie 2004; McKinney 2005; Hawley & Krolik 2006; Beckwith et al. 2008a, 2008b). Geometrically thin accretion disks (Shakura & Sunyaev 1973) are many orders of magnitude more luminous than RIAFs (e.g., Sikora et al. 2007; Jones et al. 2016) and thought to be responsible for most of the black hole growth. Over time, it became possible for the simulations to resolve not only the small thickness of thin disks but also the smaller length scales generated by the magnetized turbulence (Shafee et al. 2008; Penna et al. 2010; Noble et al. 2010), which is powered by the magnetorotational instability (MRI; Balbus & Hawley 1991).

However, recently, evidence has emerged that much higher resolutions than currently achievable by state-of-the-art simulations (e.g., $N_r \times N_\theta \times N_\phi = 256^3$ cells, where N_i is the number of cells in the i th direction) are needed to properly resolve disk turbulence in global simulations of magnetized black hole accretion. Namely, a key GRMHD code comparison project (Porth et al. 2019) found that even the resolution of 384^3 cells is insufficient to reach convergence in the critical parameters such as the value of the effective α -viscosity generated by the magnetized turbulence. To make matters worse, typically the timescales covered by GRMHD simulations fall short by order (s) of magnitude compared to observationally interesting timescales, e.g., the timescale to generate and advect large-scale poloidal magnetic fields from large radii (Liska et al. 2020), to propagate active galactic nucleus (AGN) jets to galaxy scales (e.g. McKinney 2006; Chatterjee et al. 2019; Lalakos et al. 2022), and to evolve tidal disruption events for a fallback time (e.g., Shiokawa et al. 2015; Curd & Narayan 2019; Andalman et al. 2022).

Observations and theoretical arguments furthermore suggest that accretion systems lack symmetries often taken for granted in GRMHD simulations. For instance, an accretion disk is typically misaligned with respect to the black hole spin axis (e.g., Hjellming & Rupen 1995; Greene et al. 2001; Volonteri et al. 2005; King et al. 2005; Caproni et al. 2006, 2007), but this is often not taken into account for expedience. Such a tilt



Original content from this work may be used under the terms of the [Creative Commons Attribution 4.0 licence](#). Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.

will, however, significantly change the disk dynamics and the black hole spin evolution (e.g., Bardeen & Petterson 1975; Papaloizou & Pringle 1983; Ivanov & Illarionov 1997; Ogilvie 1999; Lubow et al. 2002; Nixon et al. 2012). In fact, the dragging of inertial frames by the spinning black holes causes tilted disks to precess, which can lead to interesting periodic time variability (e.g., Fragile & Anninos 2005; Fragile et al. 2007; Lodato & Price 2010; Nixon et al. 2012; Nealon et al. 2015). To understand its potential connection to quasiperiodic oscillations (QPOs; Stella & Vietri 1998; Ingram et al. 2009, 2016; Kalamkar et al. 2016; Miller-Jones et al. 2019) observed in the light curves of X-ray binaries (XRBs; e.g., van der Klis et al. 1985; van der Klis 2006), and whether similar variability can be present in AGNs, requires even costlier simulations with even longer runtimes. If such simulations were possible, QPOs could possibly be exploited to independently verify black hole spin measurements and provide unique constraints on the disk geometry (e.g., Fragile et al. 2007; Nixon et al. 2012; Liska et al. 2021).

Indeed, the extremely high resolution and long runtime required to tackle these problems is prohibitive, limiting our ability to understand black hole accretion and its effect, or feedback, on the surrounding ambient medium. The simulation cost scales very steeply with the thickness of the disk, as $(h/r)^{-5}$ with an adaptive mesh (and as $(h/r)^{-6}$ without an adaptive mesh; Section 3.4). For example, to go from a typical thick, $h/r \sim 0.3$, low-luminosity accretion disk (Narayan & Yi 1994) to a typical thin, luminous quasar disk that is about an order of magnitude thinner, $h/r \sim 0.03$ (Shakura & Sunyaev 1973), and misaligned relative to the black hole by a typical angle of 1 rad, would require an order-of-magnitude increase in resolution in every dimension to fully capture the three-dimensional structure of MRI turbulence. This would, without adaptive and static mesh refinement (SMR; Section 3.3), require a factor $\sim 10^3$ more cells and a factor ~ 30 more time steps (Section 3.5) to run the simulation for the same amount of time. However, the simulation would need to be run for ~ 100 times longer to capture the viscous time, $t_{\text{visc}} \propto (h/r)^{-2}$, of the very thin accretion disk (Shakura & Sunyaev 1973). All together, a thin, tilted disk would increase the cost of a simulation by a factor of 10^5 – 10^6 , requiring orders of magnitude more computational power than currently feasible. It is therefore one of the most urgent challenges to improve the performance of GRMHD codes and adaptive mesh refinement (AMR) frameworks to bridge this performance gap.

One interesting way of accelerating MHD simulations is to use special hardware such as graphics processing units (GPUs). Since GPUs dedicate a much larger fraction of their transistors to raw floating point power compared to central processing units (CPUs), they can be much faster. However, writing efficient code on GPUs is challenging because GPUs require a high level of parallelism and have relatively small caches and low memory bandwidth compared to CPUs. In recent years several groups have utilized GPU acceleration in astrophysical MHD codes. For example, K-ATHENA (Grete et al. 2021), a GPU-accelerated version of Athena++ (Stone et al. 2020), achieves a factor ~ 10 speedup on an NVIDIA V100 GPU compared to a 20-core Intel Skylake CPU. The same authors have also demonstrated excellent scalability on OLCF Summit, the largest GPU cluster in production at the time of writing. Similar speedups (factor ~ 10) were obtained by the GAMER-2 MHD code (Schive et al. 2018) and the HORIZON (Zink 2011)

and cuHARM (Bégué et al. 2022) GRMHD codes. While a factor ~ 10 speedup makes existing simulations faster, it is insufficient to attack black hole accretion in the challenging regimes discussed above.

To tackle this problem, we adopt a multifaceted approach that uses (i) a highly optimized code capable of running on both CPUs and GPUs and (ii) a custom-built advanced AMR framework. Namely, we describe various optimizations that we implemented in our new GRMHD code H-AMR (pronounced “hammer”) that speed up GRMHD simulations by 2–5 orders of magnitude for the especially challenging problems discussed above. In Section 2 we introduce our code, and in Section 3 we describe our algorithmic advances—3D AMR framework and local adaptive time stepping (LAT)—that provide the bulk of H-AMR’s algorithmic speedup. In Section 3 we describe hardware-specific optimizations on both CPUs and GPUs. In Section 4 we show benchmarks for various generations of CPUs and GPUs, and in Section 4 we present the largest-ever 3D GRMHD simulation, including the scaling tests, run on OLCF Summit. We summarize our results and provide an outlook for the future development of H-AMR in Section 8.

2. Numerical Scheme

H-AMR originally branched out from an open-source HARMPI code (Tchekhovskoy 2019), which derives from the publicly available HARM2D code (Gammie et al. 2003; Noble et al. 2006). H-AMR solves the ideal GRMHD equations in the conservative form. These do not explicitly include any resistive or viscous terms. Instead, H-AMR relies on numerical dissipation from Riemann solvers to arrive at a physical solution when dissipation is necessary. This is a reasonable approximation since dissipation typically occurs on spatial scales much smaller than what can be resolved numerically in global GRMHD simulations.

Recently, H-AMR has also incorporated a radiation module (e.g., Liska et al. 2022) that uses the M1 approximation (e.g., Levermore 1984; McKinney et al. 2013) and can take into account thermal decoupling between ions and electrons (e.g., Ressler et al. 2015). We leave the description and validation of our radiation module to a future publication.

2.1. Riemann Solvers, Spatial Reconstruction, and Time Integration

H-AMR uses a finite-volume shock-capturing Godunov-based HLLC Riemann solver (Harten et al. 1983). To calculate the HLLC fluxes, we use an approximation for the fast-wave speed, v_f (e.g., Gammie et al. 2003). This overestimates v_f by up to a factor 2 in rare circumstances, which results in additional numerical dissipation, but allows for an analytic solution. Although H-AMR has the option to solve for the full dispersion relation, we do not describe this feature here since it did not significantly increase the accuracy. In addition to the two-wave HLLC solver, we have implemented the three-wave HLLC (Mignone & Bodo 2005, 2006) and five-wave HLLD (Mignone et al. 2009) solvers by transforming the Riemann problem into a locally flat Minkowski frame (Pons et al. 1998; Antón et al. 2006; White et al. 2016). However, since these less diffusive solvers can produce numerical artifacts in highly magnetized regions, we will report on their implementation once these issues are fully resolved.

If intercell fluxes are treated as Riemann problems without any spatial reconstruction, this leads to first-order spatial accuracy. However, in smooth flows it is possible to reconstruct the left and right states of the Riemann problem to achieve a higher order of convergence. In H-AMR we use piecewise parabolic, second-order-accurate spatial reconstruction of “primitive” cell variables (PPM; Colella & Woodward 1984) on cell faces with an optional shock flattener. Our implementation of PPM does not take into account nonuniform grid spacing (e.g., Mignone 2014), but we are considering this for future work. In the GRMHD community-wide code comparison project (Porth et al. 2019) we found PPM to be significantly more accurate than other second-order reconstruction schemes (MINMOD and MC; LeVeque 2002).

The conserved quantities (U) are evolved in time by advancing their state from t^n to $t^{n+1/2}$ in a predictor step and then using the fluxes calculated at $t^{n+1/2}$ to advance the state from t^n to t^{n+1} in a corrector (full) step. The time step for our midpoint method is limited by the fast-wave speed in the coordinate frame through the Courant et al. (1928) condition. We set $\Delta t = C_0/(1/\Delta t^1 + 1/\Delta t^2 + 1/\Delta t^3)$. Here $\Delta t^i = \Delta x^i/v_f$ is the time it takes a fast wave to cross a cell of size Δx^i in the i th direction and C_0 is the Courant factor that we typically set to $C_0 \sim 0.7\text{--}0.9$. Our time integration is similar to the VL2 scheme described in Stone & Gardiner (2009), but we use spatial reconstruction of variables for both the predictor and corrector steps. This formally does not guarantee that the scheme is total variation diminishing (TVD), but in practice it produces more accurate results. In the future, we plan to explore higher-order time integration schemes that are strong stability preserving, including third- and fourth-order Runge–Kutta schemes. These might provide additional stability that will be necessary for less diffusive Riemann solvers (e.g., Mignone et al. 2009) and higher-order spatial reconstruction schemes (e.g. Tchekhovskoy et al. 2007).

2.2. Coordinate System and Metric

Since H-AMR solves equations of motion in a covariant form, any smoothly varying coordinate system (and metric) can be chosen. In this work the grid is uniform in modified spherical coordinates that are defined as $x^0 \equiv t$, $x^1 \equiv \log r$, $x^2 \equiv \theta$, $x^3 \equiv \varphi$. This grid extends from an inner spherical radius R_{in} , which is located just inside the event horizon, out to an outer radius R_{out} . We choose R_{in} such that there are at least $\gtrsim 5$ cells between R_{in} and the event horizon and usually set $R_{\text{out}} = 10^4 r_g$ or larger. This ensures that both of the radial boundaries do not affect the physical evolution of the system.

The warping of spacetime is described in H-AMR by a metric tensor $g_{\mu\nu}$ with lapse $\alpha = 1/\sqrt{-g^{00}}$, shift $\beta^i = g^{0i}$, and determinant $g = \text{Det}(g_{\mu\nu})$. We use (μ, ν) as indices for four-dimensional objects and (i, j) as indices for three-dimensional objects. $\Gamma = -u^\mu n_\mu = \alpha u^i$ is the Lorentz factor corresponding to 4-velocity u^μ as measured by the zero angular momentum observer (ZAMO) with 4-velocity $n_\mu = (-\alpha, 0, 0, 0)$. The spatial projection tensor γ_ν^μ is defined as $\gamma_\nu^\mu = g_\nu^\mu + n^\mu n_\nu$, and the four-dimensional Levi-Civita symbol is $\epsilon^{\mu\nu\lambda\delta}$.

In this work we use a Kerr spacetime (unless stated otherwise). We first calculate the Kerr metric tensor analytically in Kerr–Schild coordinates and then transform it numerically to give $g_{\mu\nu}$ in internal coordinates. The corresponding Christoffel symbols, $\Gamma_{\mu\nu}^\zeta$, are calculated numerically by taking finite differences of the metric in internal coordinates.

2.3. Primitive Variables

We express the conservation laws that H-AMR evolves (Section 2.4) in terms of fluid-frame gas density ρ , internal energy density u_g , gas pressure p_g , fluid-frame magnetic 4-vector b^μ , and fluid 4-velocity u^μ . These relate to the primitive variables ρ , u_g , relative 4-velocity \tilde{u}^i , and coordinate frame magnetic field B^i as follows (e.g., Noble et al. 2006):

$$b^i = \frac{B^i + b^i u^i}{u^0} \quad (1)$$

$$b^t = g_{i\mu} B^i u^\mu \quad (2)$$

$$\tilde{u}^i = \gamma_\mu^i u^\mu = u^i + \frac{\beta^i}{\alpha} \Gamma. \quad (3)$$

The relative 4-velocity \tilde{u}^i is numerically convenient because it guarantees that $u^\mu u_\mu = -1$ for any value of \tilde{u}^i . In H-AMR we store the primitive variables at cell centers. In this manuscript we adopt an ideal gas equation of state with $p_g = (\gamma - 1)u_g$ and Lorentz–Heaviside units, $G = M = c = 1$. The stress energy tensor T_ν^μ , Faraday tensor $\mathcal{F}^{\mu\nu}$, dual of the Faraday tensor $\mathcal{F}^{*\mu\nu}$, and entropy tracer κ are given by

$$T_\nu^\mu = (\rho + u_g + p_g)u^\mu u_\nu + g_\nu^\mu \left(p_g + \frac{1}{2}b^2 \right) - b^\mu b_\nu \quad (4)$$

$$\mathcal{F}^{\mu\nu} = -\sqrt{-g} \epsilon^{\mu\nu\lambda\delta} u_\lambda b_\delta \quad (5)$$

$$\mathcal{F}^{*\mu\nu} = b^\mu u^\nu - b^\nu u^\mu \quad (6)$$

$$\kappa = \frac{p_g}{\rho^\gamma}. \quad (7)$$

The coordinate frame magnetic field B^i and electric field E^i are related to the (dual of the) Faraday tensor as follows:

$$B^i = \mathcal{F}^{*it} \quad (8)$$

$$E^i = \mathcal{F}^{it}. \quad (9)$$

2.4. Conservation Laws

H-AMR evolves the mass (Equation (10)), energy (Equation (11)), momentum (Equation (12)), induction (Equation (13)), and entropy (Equation (14)) equations. The derivation of these equations from the relativistic Boltzmann equation can be found in standard textbooks, including Misner et al. (1973) and Rezzolla & Zanotti (2013). Here we express these equations in covariant form where ∇_μ denotes a covariant derivative:

$$\nabla_\mu(\rho u^\mu) = 0 \quad (10)$$

$$\nabla_\mu(T_t^\mu) = 0 \quad (11)$$

$$\nabla_\mu(T_i^\mu) = 0 \quad (12)$$

$$\nabla_\mu(\mathcal{F}^{*\mu\nu}) = 0 \quad (13)$$

$$\nabla_\mu(\kappa \rho u^\mu) = 0. \quad (14)$$

For numerical reasons, we subtract the mass conservation equation from the energy equation, i.e., H-AMR actually evolves $\nabla_\mu(T_t^\mu + \rho u^\mu) = 0$ instead of Equation (11).

2.5. Discretization

Figure 1 shows how we evolve the conservation laws (Equations (10)–(14)) on a numerical grid with cell indices (i, j, k) and uniform cell sizes of $(\Delta x^1, \Delta x^2, \Delta x^3)$. We absorb a

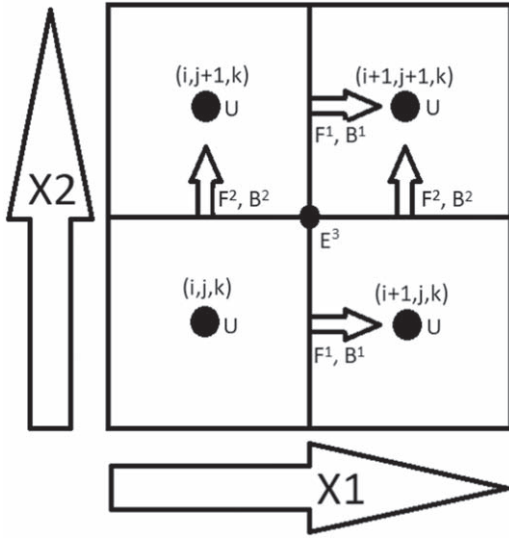


Figure 1. An illustration of a cell in the (x^1, x^2) -plane in H-AMR. The conserved quantities U are stored at cell centers, the fluxes $F^{1,2}$ and staggered magnetic field components $B^{1,2}$ are stored at cell faces, and the electric fields E^3 are stored at cell edges. The corresponding metric components $g_{\mu\nu}$ are stored for cell centers, cell faces, and cell corners. The conserved states are calculated directly from the primitive variables. The fluxes are calculated from the reconstructed primitive variables in a Riemann solver. The electric fields are calculated (in their simplest form) by taking an average of the neighboring fluxes. Typically, H-AMR modifies the calculation of the electric fields with a velocity upwinding routine as described in Gardiner & Stone (2005).

geometric factor, $\sqrt{-g}$, into the definitions for the volume-integrated conserved state $\bar{U} = U\sqrt{-g}$, volume-integrated source terms $\bar{Q} = Q\sqrt{-g}$, and area-integrated fluxes in the j th direction $\bar{F}^j = F^j\sqrt{-g}$. This factor is calculated at the center of each cell for cell-centered quantities and at the center of each face for face-centered quantities in internal coordinates. This is an easy and fast method but slightly less accurate (albeit still second-order) than explicitly integrating the volume and area over each cell to account for nonuniform grid spacing (e.g., Porth et al. 2017).

In H-AMR, the vector $U = (\rho u^t, T_i^t, T_i^t, B^i, \kappa\rho u^t)$ is calculated directly at cell centers from the primitive variables, while $F^j = (\rho u^j, T_i^j + \rho u^t, T_i^j, b^j u^i - b^i u^j, \kappa\rho u^j)$ is calculated at cell faces from the reconstructed primitive variables using a Riemann solver. The equations of motion take the conservative form $\frac{\partial U}{\partial t} + \frac{\partial F^j}{\partial x^j} = Q$. With these definitions we can write down the equations of motion in conservative form for the full time step:

$$\begin{aligned} \frac{(\bar{U}_{i,j,k}^{n+1} - \bar{U}_{i,j,k}^n)}{\Delta t} = & \\ - \frac{(\bar{F}^1)_{i+1/2,j,k}^{n+1/2} - (\bar{F}^1)_{i-1/2,j,k}^{n+1/2}}{\Delta x^1} - \frac{(\bar{F}^2)_{i,j,k+1/2}^{n+1/2} - (\bar{F}^2)_{i,j,k-1/2}^{n+1/2}}{\Delta x^2} & \\ - \frac{(\bar{F}^3)_{i,j,k+1/2}^{n+1/2} - (\bar{F}^3)_{i,j,k-1/2}^{n+1/2}}{\Delta x^3} + \bar{Q}_{i,j,k}^{n+1/2}. & \end{aligned} \quad (15)$$

In the simplest case, the source term \bar{Q} accounts for the warping of the spacetime and grid (e.g., $\bar{Q}_i = \Gamma_{\nu i}^{\zeta} T_{\zeta}^{\nu} \sqrt{-g}$ for the momentum equation), but it may also include physical processes such as nuclear heating, gas–radiation interactions, and Coulomb coupling (e.g., Liska et al. 2022).

According to Stokes’s theorem, to guarantee the divergence-free evolution of the magnetic field, the induction equation is evolved on a staggered mesh (e.g., Evans & Hawley 1988). Here, the face-centered magnetic field is advanced in time by taking finite differences in space of the edge-centered electric fields, as we show in Figure 1:

$$\begin{aligned} \frac{(\bar{B}^1)_{i-1/2,j,k}^{n+1} - (\bar{B}^1)_{i-1/2,j,k}^n}{\Delta t} & \\ = \frac{(\bar{E}^2)_{i-1/2,j,k+1/2}^{n+1/2} - (\bar{E}^2)_{i-1/2,j,k-1/2}^{n+1/2}}{\Delta x^3} & \\ - \frac{(\bar{E}^3)_{i-1/2,j+1/2,k}^{n+1/2} - (\bar{E}^3)_{i-1/2,j-1/2,k}^{n+1/2}}{\Delta x^2} & \end{aligned} \quad (16)$$

$$\begin{aligned} \frac{(\bar{B}^2)_{i,j-1/2,k}^{n+1} - (\bar{B}^2)_{i,j-1/2,k}^n}{\Delta t} & \\ = \frac{(\bar{E}^3)_{i+1/2,j-1/2,k}^{n+1/2} - (\bar{E}^3)_{i-1/2,j-1/2,k}^{n+1/2}}{\Delta x^1} & \\ - \frac{(\bar{E}^1)_{i,j-1/2,k+1/2}^{n+1/2} - (\bar{E}^1)_{i,j-1/2,k-1/2}^{n+1/2}}{\Delta x^3} & \end{aligned} \quad (17)$$

$$\begin{aligned} \frac{(\bar{B}^3)_{i,j,k-1/2}^{n+1} - (\bar{B}^3)_{i,j,k-1/2}^n}{\Delta t} & \\ = \frac{(\bar{E}^1)_{i,j+1/2,k-1/2}^{n+1/2} - (\bar{E}^1)_{i,j-1/2,k-1/2}^{n+1/2}}{\Delta x^2} & \\ - \frac{(\bar{E}^2)_{i+1/2,j,k-1/2}^{n+1/2} - (\bar{E}^2)_{i-1/2,j,k-1/2}^{n+1/2}}{\Delta x^1}. & \end{aligned} \quad (18)$$

We approximate the edge-centered electric field by taking the average of the face-centered HLL fluxes ($\bar{F}^j(B^i) = F^{*ji}\sqrt{-g}$) over four neighboring cells:

$$\begin{aligned} (\bar{E}^1)_{i,j-1/2,k-1/2} = 0.25[\bar{F}^3(B^2)_{i,j,k-1/2} + \bar{F}^3(B^2)_{i,j-1,k-1/2} & \\ - \bar{F}^2(B^3)_{i,j-1/2,k-1} - \bar{F}^2(B^3)_{i,j-1/2,k}] & \end{aligned} \quad (19)$$

$$\begin{aligned} (\bar{E}^2)_{i-1/2,j,k-1/2} = 0.25[\bar{F}^1(B^3)_{i-1/2,j,k-1} + \bar{F}^1(B^3)_{i-1/2,j,k} & \\ - \bar{F}^3(B^1)_{i,j,k-1/2} - \bar{F}^3(B^1)_{i-1,j,k-1/2}] & \end{aligned} \quad (20)$$

$$\begin{aligned} (\bar{E}^3)_{i-1/2,j-1/2,k} = 0.25[\bar{F}^2(B^1)_{i-1,j-1/2,k} + \bar{F}^2(B^1)_{i,j-1/2,k} & \\ - \bar{F}^1(B^2)_{i-1/2,j,k} - \bar{F}^1(B^2)_{i-1/2,j-1,k}]. & \end{aligned} \quad (21)$$

In addition, H-AMR upwinds these electric fields to add extra dissipation, which prevents unphysical oscillations in multi-dimensional flows (e.g., Gardiner & Stone 2005; White et al. 2016).

To convert the face-centered magnetic field components to cell-centered magnetic field components, we take volume-weighted averages in internal coordinates:

$$(\bar{B}^1)_{i,j,k} = \frac{(\bar{B}^1)_{i-1/2,j,k} + (\bar{B}^1)_{i+1/2,j,k}}{2} \quad (22)$$

$$(\bar{B}^2)_{i,j,k} = \frac{(\bar{B}^2)_{i,j-1/2,k} + (\bar{B}^2)_{i,j+1/2,k}}{2} \quad (23)$$

$$(\bar{B}^3)_{i,j,k} = \frac{(\bar{B}^3)_{i,j,k-1/2} + (\bar{B}^3)_{i,j,k+1/2}}{2}. \quad (24)$$

2.6. Variable Inversion

In H-AMR, the conversion of conserved variables U to primitive variables p is performed using a two-dimensional Newton–Raphson root-finding method (Noble et al. 2006) or Aitken acceleration scheme (Hamlin & Newman 2013; Newman & Hamlin 2014). To provide a backup inversion method for primitive variable recovery if the energy-based inversion method(s) fail, H-AMR also evolves the entropy equation (Equation (14)), which is then used in place of the energy equation (Equation (11)) to recover the primitive variables (Noble et al. 2009). To our experience, most of the inversion failures occur in highly magnetized regions such as jets, where the lack of shock heating and magnetic reconnection does not have significant effects on the dynamics since u_g is subdominant (e.g., Chatterjee et al. 2019). In fact, evolving the entropy gives us the conservative (lower limit) estimate of the internal energy. Note that since the total entropy per particle is given by $S = \frac{1}{\gamma-1} \ln \kappa$, evolving $U = \kappa \rho u^t$ instead of $U = S \rho u^t$ does not conserve entropy to machine precision. However, it avoids numerical issues (e.g., Sadowski et al. 2016) associated with evolving logarithms, which, to our experience, makes the scheme more robust.

2.7. Density and Internal Energy Floors

Magnetic field lines in the polar regions (those that thread the black hole’s event horizon) become devoid of matter: the gas is either expelled outward or consumed by the black hole. Because GRMHD equations are vacuum-phobic, H-AMR artificially injects mass and internal energy in the drift frame of the jet if the density or internal energy drops below a certain threshold (Ressler et al. 2017). This avoids both a runaway in velocity, which can occur when mass is inserted in the fluid frame, and a drag on the field lines, which can occur when mass is inserted in a ZAMO frame (McKinney et al. 2012). For this particular work we enforce $b^2/\rho < 50$, $b^2/u_g < 750$, $\rho > 10^{-7}/r^2$, and $u_g > 10^{-9}/r^{10/3}$ for the gas density ρ and internal energy u_g .

3. Numerical Optimizations

H-AMR is written in C and triple parallelized: (i) CUDA or OpenCL does most of the computations on GPUs or AVX-accelerated CPUs, (ii) OpenMP routines handle communication and gridding, and (iii) nonblocking MPI handles the transfer of boundary cells between nodes. NVLINK is used for transfers between GPUs on a single node and GPU-DIRECT for MPI transfers between GPUs on different nodes. We found that this improves the performance by up to 20% on large GPU clusters.

H-AMR employs limited communication–computation overlap by (un)packing the data into send/receive buffers and applying boundary conditions in parallel with (nonblocking) MPI send/receive calls for other SMR/AMR blocks. This is made possible by using a separate CUDA stream for each block. H-AMR also features fully nonblocking MPI-parallelized input/output (I/O), allowing substantial overlap between computation and data transfer. Data are stored in binary format. Since individual dump files can reach 10^2 – 10^3 GB in size, we use OpenMP parallelized C kernels coupled to a Python script

to postprocess the data. These analysis tools also allow the user to load in data at reduced resolutions for postprocessing.

Memory to store all variables (primitive, conserved, fluxes, etc.) is dynamically allocated on both the CPU and GPU at the start of each run. During each (de)refinement and load-balancing step, C pointers are used to keep track of the physical location in memory of each SMR/AMR block. This avoids needing to allocate and deallocate large chunks of memory.

3.1. GPU Acceleration in CUDA

H-AMR is the first GPU-accelerated GRMHD code that has run on thousands of GPUs. While CPUs spend the largest portion of their silicon on control logic and caches, GPUs spend most of their silicon on floating-point power. This presents unique challenges in optimizing complex MHD codes (e.g., Grete et al. 2021). Hence, H-AMR follows the philosophy of keeping the code as simple as possible, avoiding abstract concepts such as classes, extra function layers, and three-dimensional arrays, which makes it easy for the CUDA compiler to generate highly efficient code. In comparison to the GPU-accelerated (GR)MHD codes KHARMA (B. Prather et al. 2022, in preparation) and K-ATHENA (Grete et al. 2021), which use KOKKOS (Carter Edwards et al. 2014), we chose to implement the GRMHD equations directly into NVIDIA’s CUDA. KOKKOS is a performance portability toolkit that can compile existing C code on various architectures. While KOKKOS increases the performance portability of a code across various architectures, it is unlikely to be able to match the performance of a highly optimized CUDA code that makes extensive use of NVIDIA-specific optimizations.

It was challenging to develop the GPU version of H-AMR owing to the large code size (over 40,000 lines of C code and 10,000 lines of CUDA kernel code). This made it difficult to fit all necessary instructions and variables into the GPU’s register space. Failing to fit the code into the available register space would slow down the performance by an order of magnitude, because the GPU would need to use the much slower GPU RAM to read and store temporary data. Kernels in many conventional codes are rather small and do not fill up the GPU register space easily. GRMHD codes, however, need to store metric components of curved spacetime of a black hole and require nonlinear root finding to convert conserved to primitive quantities (e.g., Noble et al. 2006). For this reason, fitting the GRMHD equations required many profiler-informed optimizations based on an iterative trial-and-error approach. It was, for example, beneficial to split the code into extra kernels to recalculate certain quantities instead of storing them in temporary variables using register space. This profiler-based optimization strategy has substantially increased the performance of H-AMR since 2016. The first 2016 version of H-AMR was memory bandwidth limited because of significant register spillover to global memory. Eventually we reduced the required memory bandwidth and increased H-AMR’s performance fivefold by rewriting various functions to fit into the available register space. We do not make extensive use of shared memory (e.g., Bégue et al. 2022), as we found that this slows down computation compared to a more efficient use of register space. Shared memory is only used in H-AMR to calculate the minimum time step in each mesh-block through parallel reduction.

There exists a trade-off, different for each GPU generation, between making GPU kernels too big and complex to fit into the available register space and having too many small kernels with the added overhead. On the previous generation of GPUs (NVIDIA Kepler architecture) it was, for example, beneficial to split the code into extra kernels to recalculate certain quantities instead of storing them in temporary variables using register space. On the present generation of GPUs (NVIDIA Pascal/Volta/Ampere architectures) it is beneficial to make some kernels a bit bigger, due to the larger register space. We achieve best performance when we split a time step into ~ 7 separate kernels.

All functions necessary to evolve the state of a cell from t^n to t^{n+1} are GPU accelerated. These include reconstruction routines, calculation of HLL fluxes, conserved-to-primitive variable inversion, setting of density/internal energy floors, and exchange of boundary/ghost cells. The CPU handles all the necessary logic to manage the transfer of boundary cells, fluxes, and electric fields between the neighboring blocks and performs the (de)refinement and load-balancing steps, including the divergence-free prolongation and restriction of magnetic fields (Balsara 2001). We have summarized the parallelization strategy of H-AMR in a code diagram (Figure 2).

3.2. AVX Acceleration in OpenCL

In addition to the “normal” OpenMP+MPI C-based CPU code and GPU-accelerated CUDA code, H-AMR maintains a legacy OpenCL branch that can make use of AVX vectorization on CPUs. This effectively leverages the 512-bit-wide AVX vector registers in the latest generation of Intel Skylake CPUs. Without any CPU-specific optimizations to our GPU kernels (e.g., with the CUDA kernels only ported to OpenCL), H-AMR achieves $\sim 12\%$ saturation of the CPU’s (16 Core AVX-512 2.8 GHz Skylake) theoretical FP64 throughput and gains a factor ~ 3 speedup compared to standard OpenMP code.

3.3. Adaptive Mesh Refinement

AMR (see Berger & Colella 1989; Balsara 2001) allows H-AMR to focus the resolution on regions of interest. While not bringing significant advantages for simulating thick accretion disks, which span most of the computational domain, AMR can speed up computations requiring high resolution of small-scale features that fill a small fraction of the computational volume. For instance, for problems such as thin accretion disks (Figure 3), tidal disruption events, and large-scale collimated jets, AMR can reduce the number of required cells by orders of magnitude.

AMR can also speed up the calculations in another way: by *reducing* the spatial resolution in the blocks closest to the black hole and thereby increasing the time step. Such a reduction in resolution is feasible, because GRMHD modeling has consistently shown that logarithmically spaced spherical grids typically sufficiently resolve the turbulence closest to the black hole (e.g., within two event horizon radii), while resolving the outer regions in the accretion disk and relativistic jets remains a challenge (e.g., McKinney 2006; Liska et al. 2019a; Porth et al. 2019).

We designed the AMR framework in H-AMR from the ground up for performance and scalability. We make use of an oct-tree-based approach, where any parent block can be split into two, four, six, or eight child blocks (see Section 3.4).

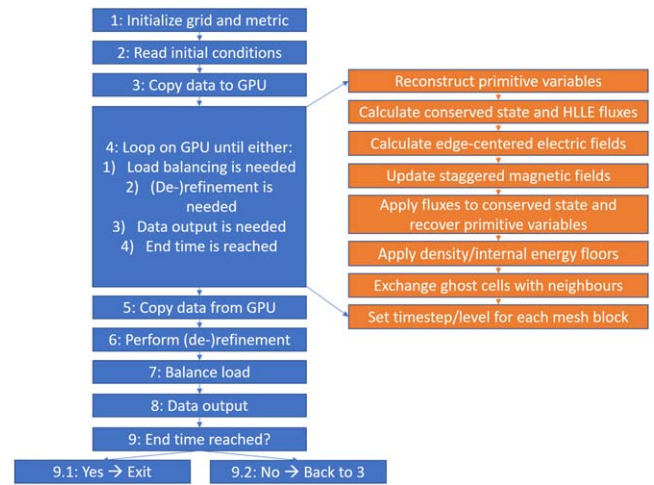


Figure 2. A code diagram of H-AMR. On the left (blue) are all tasks that are performed using OpenMP parallelized C functions. On the right (orange) are all tasks that are performed using GPU-accelerated CUDA kernels. Practically all of the computation is GPU accelerated, and the nonaccelerated parts in H-AMR typically do not form a performance bottleneck.

Compared to a patch-based AMR approach, we do not need to evolve the underlying parent blocks for each refined layer and, instead, can directly transfer boundary cells from neighboring coarse to fine layers of the grid (and vice versa). This drastically reduces the required internode MPI bandwidth, which presents the main performance bottleneck for our simulations. Since the relative internode MPI bandwidth is expected to decrease for the next generation of GPU clusters, oct-tree-based AMR is an attractive approach for the next generation(s) of supercomputers.

Since every problem is different, H-AMR allows the user to implement an arbitrary refinement criterion. For example, Liska et al. (2019a, 2021) used a cutoff on the maximum density in each block as the refinement criterion. Blocks satisfying the chosen refinement criterion will be refined to the highest AMR level allowed for a problem. Neighboring blocks (adjacent along the faces and edges) for each refined block, which do not satisfy the refinement criterion, are refined to one AMR level below that of the refined block. These nesting conditions prevent resolution jumps by factors greater than 2 along block faces and edges.

We use prolongation and restriction operators similar to those described in Berger & Olinger (1984) and Berger & Colella (1989) for cell-centered variables and Balsara (2001) for face-centered variables. These operators upscale/downscale the resolution of the grid during (de)refinement steps and reconstruct primitive variables in ghost cells at coarse–fine boundaries. During prolongation, we use MINMOD-limited gradients calculated on the coarse mesh to upscale the resolution of the variables to a finer mesh. By using a more diffusive MINMOD limiter, we prevent numerical issues at refinement boundaries associated with applying a 1D reconstruction scheme to a 3D stencil (see also Stone et al. 2020). During restriction, we take the volume- or area-weighted averages on the fine mesh to reduce the resolution of the variables on the coarse mesh. Prolongation and restriction can be performed on either the primitive variables or conserved quantities. The former is faster and more robust (e.g., does not lead to states with $v > c$), while the latter is energy and angular momentum conserving up to machine precision. In general, if the refinement criterion is set such that no (de)refinement

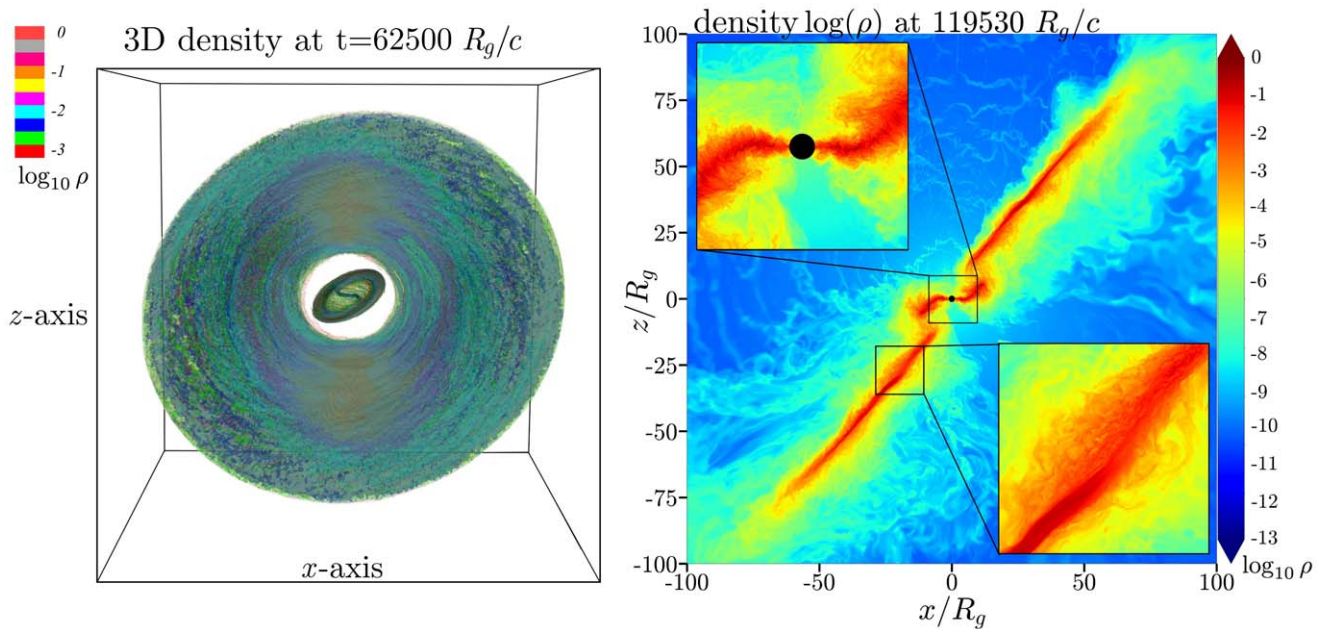


Figure 3. GRMHD simulation of a thin accretion disk of aspect ratio $h/r = 0.02$ initially threaded by a purely toroidal magnetic field and tilted by 65° relative to the (horizontal) equatorial plane of the black hole. The frame dragging by the spinning black hole rips the accretion disk apart into inner and outer subdisks as can be seen in the rendering of density isocontours in the left panel. The black hole spin points along the positive z -direction. The right panel shows a vertical slice through the density. The top left zoom-in inset shows that the inner regions of the accretion flow reorient themselves parallel to the black hole midplane as it undergoes Bardeen–Peterson (Bardeen & Petterson 1975) alignment, the first demonstration of this effect in GRMHD in the absence of large-scale poloidal magnetic field (see also Liska et al. 2019a, 2021), solving a 40 yr old problem (Bardeen & Petterson 1975). The bottom right inset illustrates that the small-scale turbulent structure of the accretion flow is very well resolved. An animation of this figure is available online. A higher-resolution version is also available as a YouTube movie (www.youtube.com/watch?v=rIOjKUfzcvI). The animation runs from $t \sim 10,000 r_g/c$ to $t \sim 138,000 r_g/c$ and has a real-time duration of 104 s.

(An animation of this figure is available.)

happens in the area of interest, we recommend performing prolongation and restriction directly on the primitive variables.

In addition, as described in Berger & Colella (1989), we replace at the end of a time step the fluxes on the coarser mesh with the area-averaged fine mesh fluxes and edge-averaged electric fields. This is necessary since the sum of the fluxes on the fine mesh might not match to machine precision the flux on the coarse mesh. This refluxing guarantees conservation of mass, energy, and angular momentum, in addition to the divergence-free evolution of magnetic fields. While refluxing of cell-centered quantities only involves six faces, refluxing of the edge-centered electric fields involves, in addition to the six faces, 12 cell edges. In general, we pick the cell edge at the highest refinement and time-stepping level as being “dominant.” The electric fields on neighboring edges are subsequently synchronized with the “dominant” edge. To make sure that this constrained-transport algorithm is implemented correctly, we periodically verify consistency up to machine precision between the staggered magnetic field components at cell boundaries of neighboring meshblocks.

3.4. Handling of the Polar Region

Spherical grids are perhaps the most natural and efficient grids to study accretion disks. They naturally follow the geometry of the disk rotating around the black hole. The spherical shape of the event horizon in the Kerr–Schild foliation naturally matches that of the grid’s inner boundary, $r = R_{\text{in}}$. In addition, spherical grids support a logarithmic spacing in the radial coordinate, naturally providing high resolutions close to the black hole where the timescales are short, and lowering the resolution progressively as one moves

out and the timescales become longer. This is advantageous compared to, for example, Cartesian grids, as one does not need to use many AMR layers to get the typically required 2-to-5-order-of-magnitude scale separation between inner and outer boundary of the grid. This in turn allows for the use of relatively large AMR block sizes, minimizing the number of boundary cell transfers and leading to excellent parallel scaling (Section 4).

A drawback of spherical grids is the polar coordinate singularity, which requires special treatment. We have implemented transmissive boundary conditions across the singularity to minimize dissipation and verified that the grid is robust for the study of tilted accretion disks and jets (see the Appendix of Liska et al. 2018 for details). Even with these improvements, spherical grids still suffer from cell “squeezing” near the pole in the azimuthal direction. This causes the Courant et al. (1928) condition (which limits the time step to the wave-crossing time of a single cell) to limit the global time step more than for a Cartesian with an equivalent resolution. Since close to the polar singularity the cell aspect ratio $r\Delta\theta/(r\sin\theta\Delta\varphi) \simeq \Delta\varphi^{-1} = N_\varphi/(2\pi)$ increases proportionally to the azimuthal resolution, the time step for a spherical grid will, empirically, be a factor $\sim N_\varphi/(2\pi)$ times smaller than that for a Cartesian grid with the same effective resolution.⁵ As a result, in 3D the cost of simulations on a spherical grid increases as $\propto N^5$ instead of the typical $\propto N^4$ for Cartesian

⁵ More precisely, the reduction factor will be 3 times smaller, since the time step in H-AMR is set as the harmonic mean of the time steps in each of the three dimensions, $1/\Delta t = 1/\Delta t_1 + 1/\Delta t_2 + 1/\Delta t_3$. Hence, for cubical cells, $\Delta t \approx (1/3)\Delta t^1 \approx (1/3)\Delta t^2 \approx (1/3)\Delta t^3$.

grids, making high-resolution (GR)MHD simulations on spherical grids particularly expensive.

There are three popular approaches to avoiding the squeezing of the cells in the φ -direction. One is the use of Cartesian grids (e.g Porth et al. 2019). The advantage of this method is that it is symmetry agnostic. Disadvantages include the added computational cost because the grid does not conform to the shape of the black hole or the orbital motion of the gas. Such grids can also require many AMR levels (e.g., eight AMR levels; Davelaar et al. 2019), which can reduce parallel scaling efficiency, a point of concern especially for GPU-based systems with a low ratio of MPI interconnect bandwidth to computation power. The second approach (e.g Stone et al. 2020) is to use SMR to derefine the grid near the pole in all three dimensions, (r , θ , φ). The lower polar resolution reduces the computational cost and is excellent for applications focused on the physics of the equatorial accretion flow. However, the larger cell size in the r - and θ -directions also makes it more difficult to resolve polar structures such as jets or tilted disks. Currently, most GRMHD codes use a third strategy that involves enlarging the innermost few cells in the θ -direction (Tchekhovskoy et al. 2011). While this approach works well at low resolutions, it becomes ineffective at higher resolutions since the relative size of the innermost cells increases proportionally to the ϕ -resolution. For example, at the grid resolution of 128^3 cells the innermost cell needs to be enlarged by a factor ~ 10 , while at the resolution of 1024^3 cells it would need to be enlarged by a factor ~ 100 . This makes grid cylindrication unsuitable for high resolutions.

3.4.1. External SMR

In H-AMR, we attempted to combine the best aspects of both of the above approaches. We use SMR to avoid the squeezing of cells in the φ -direction (Liska et al. 2018). For this, H-AMR derefines the base layer of the grid in the φ -direction within 30° of the pole. This maintains the full resolution in the r - and θ -directions, leading to a uniform cell aspect ratio (within a factor of 2) across radial shells (Figure 4). As an example, our fiducial simulation (which uses base grid with, effectively, $1728 \times 576 \times 1024$ cells; Section 4) has a φ -resolution of 256 cells for $0^\circ < \theta < 15^\circ$, 512 cells for $15^\circ < \theta < 30^\circ$, and the full 1024 cells for $30^\circ < \theta < 90^\circ$. We refer to this approach as “external SMR,” to distinguish it from “internal SMR” discussed below (see Section 3.4.2).

A similar approach to reduce the resolution near the pole is utilized in the radiation hydrodynamics FORNAX code (Skinner et al. 2019). However, to our knowledge, H-AMR is unique among (GR)MHD codes in its ability to combine one-dimensional SMR with three-dimensional AMR. When refined with AMR, all parent blocks in our SMR-enhanced base grid are split into eight child blocks, except the blocks touching the polar axis. These blocks are split into six child blocks, where the two child blocks along the pole maintain their parent’s φ -resolution and the four child blocks farther away from the pole are refined in all three dimensions. This way the cells near the pole are kept approximately cubic by the (effective) addition of a derefination layer in the φ -direction for each additional AMR layer.

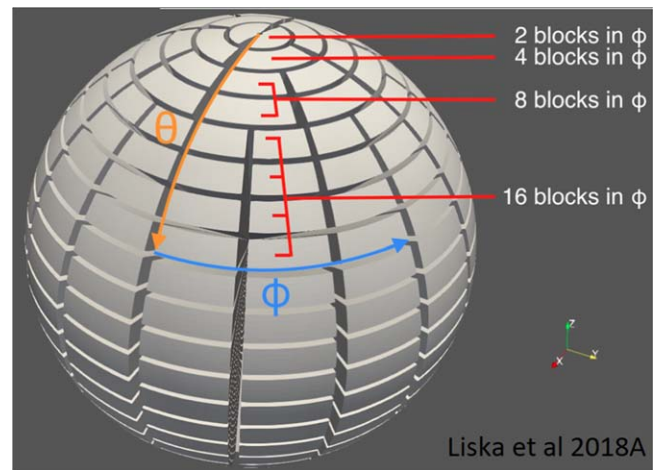


Figure 4. As previously presented in Liska et al. (2018), the spherical “base” grid in H-AMR has a varying number of blocks in φ . This prevents cell squeezing near the pole, speeding up high-resolution GRMHD simulations by up to 2 orders of magnitude without any loss in accuracy. AMR can increase the resolution of all blocks in this grid, including those adjacent to the polar axis.

3.4.2. Internal SMR

Typically, H-AMR uses external SMR to reduce the φ -resolution down to $N_\varphi \sim 64$ –256 cells near the pole. Bringing down the φ -resolution to $N_\varphi \sim 16$ cells would require two to four extra refinement layers featuring exceedingly small block sizes. This would oversaturate the available memory and MPI interconnect bandwidth, leading to a serious performance bottleneck on GPU clusters. To remedy this limitation, H-AMR also implements φ -derefination internally on the block level, which we refer to as internal SMR. Depending on the distance of a cell from the pole, internal SMR spatially averages the fluxes and conserved quantities of the cell over 2^n cells in the φ -direction within $2^{n_{\max}-n}$ cells from the pole such that the cell aspect ratio remains close to unity. Here n is the internal SMR level and n_{\max} is the number of internal SMR levels. This can reduce the φ -resolution by an additional factor 4–16, which is sufficient to completely eliminate cell “squeezing” near the pole while exploiting the computational advantages of larger-sized blocks. To maintain higher-order spatial accuracy, the spatial reconstruction method is modified accordingly to compensate for the (effectively) increased cell spacing, making internal SMR equivalent to external SMR.

3.5. Local Adaptive Time Stepping

H-AMR features an LAT. The typical approach for this in AMR codes is to use smaller time steps for higher-level AMR layers (aka hierarchical time stepping). In addition, H-AMR exploits the fact that on a logarithmic grid cell size increases with radius, even without any spatial refinement layers. Based on the local Courant et al. (1928) condition, LAT allows H-AMR to determine the time step size (in steps of a factor of 2) for each block (this includes the block’s ghost cells) independently from the block’s spatial refinement level. To guarantee second-order convergence in time, we typically evolve the coarser grid first such that we can estimate (by linear interpolation in time) the ghost cell variables on the fine grid. Subsequently, in a flux correction step (see also Section 3.3), the flux difference between the fine and course grid is applied to the coarse grid.

LAT leads to a factor of 3–10 reduction in the number of time steps, depending on the structure of the adaptive grid. This manifests itself as a factor 2–3 speedup due to extra overhead associated with parallel scaling (see Section 3.6). In general, grids with many refinement layers farther away from the black hole achieve the most benefit from LAT, while grids with no AMR only manage to reduce the number of time steps by a factor \sim few. In contrast, using hierarchical time stepping, which couples the time step to the spatial refinement level, would lead to a smaller speedup of 10%–30% since for most problems most blocks reside at the highest spatial refinement level. In addition to speeding up computations, LAT also increases the numerical accuracy by reducing the number of conserved-to-primitive variable inversions in the outer grid and thereby reducing the noise generated by the inversions (see the Appendix in Chatterjee et al. 2019). This is especially pronounced in large-scale jets, where the calculations due to the large difference between the magnetic energy density and rest mass energy density are prone to inversion errors.

3.6. Load Balancing

We use a z -order space-filling curve for load balancing in order to keep neighboring blocks in the grid physically close to each other on a cluster (e.g., on the same or neighboring nodes). Depending on the problem specifics (such as the number of spatial and temporal refinement levels) and the architecture of the cluster (fat-tree or 3D-torus), we found that changing the fastest-moving index in the space-filling curve or switching to a row-major order at the zeroth level may significantly improve performance. AMR (de)refinement is performed every 10^3 – 10^4 time steps. This is similar to one orbital period at the black hole event horizon and hence sufficient to capture the dynamical evolution of the accretion disk and jet.

We note that LAT brings about two technical difficulties. First, it significantly complicates load balancing. If one were to keep the number of time steps per GPU equal and naively follow the space-filling curve, a huge imbalance in memory consumption could occur, causing some GPUs to run out of memory. Second, it makes the synchronization of fluxes between different time levels more difficult: at every time step the conserved fluxes and electric fields need to be synchronized between the fine and coarse time levels on block faces and edges to guarantee energy/momentum conservation and the divergence-free evolution of magnetic fields (Balsara 2001). For this, by the end of each full time step the flux difference between the fine and coarse layers is added as a correction to the coarse block’s boundary cells, and thus an additional conserved-to-primitive variable inversion needs to be performed for these boundary cells. This tightly couples the fine and coarse layers, potentially leading to an unbalanced load that slows down computation. We now discuss how we alleviate both of the issues.

To prevent an imbalance in memory consumption between GPUs, H-AMR uses a load-balancing strategy that attempts to keep both the number of time steps and the number of blocks at each time level per GPU constant, while adhering as closely as possible to the utilized space-filling curve. This can be achieved by loading in on every GPU a similar number of computationally intensive (with many time steps) and non-intensive (with few time steps) blocks by load-balancing the grid separately for each time level and using the next time level

to “fill” up the imbalance created during load balancing of the previous time level. To prevent GPUs from running out of memory during, e.g., (de)refinement steps (each 16 GB GPU can only fit a grid of approximately $\sim 256^3$ cells), H-AMR may in rare cases also perform intermediate load-balancing steps or may reduce the number of time levels (if it determines that this indeed reduces the internode memory imbalance).

4. Performance Measurements

In this section, we describe the physical and numerical setup of the GRMHD simulation we used as a benchmark. This simulation is illustrated using volume rendering and a transverse slice through density in Figure 3. It ran in 2019 on then the world’s most powerful supercomputer, OLCF Summit, which is powered primarily by 27,648 NVIDIA V100 GPUs, each with 16 GB of HBM2 RAM. We also discuss how we measure the performance and determine the scaling efficiency of H-AMR.

4.1. Physical Setup

We consider a thin accretion disk with scale height $h/r = 0.02$ around a spinning black hole of spin $a = 0.94$ tilted by 65° . Our previous work initialized the accretion disk with a poloidal magnetic flux loop (Liska et al. 2019a, 2021). In contrast, here the disk is threaded by a toroidal magnetic field with an approximately uniform plasma $\beta \sim 10$ (Musoke et al. 2022). To our knowledge, this is the first GRMHD simulation of a thin disk in the absence of initial poloidal magnetic flux, making it extra challenging since a toroidal magnetic field needs a much higher resolution to be properly resolved (Liska et al. 2020). Such a magnetic field topology is thought to be responsible for most of the luminous black hole systems in our universe, since most of such systems lack a jet, while a poloidal magnetic field guarantees the production of a jet. We discuss the physical implications of the simulation in Section 8.2.

4.2. Numerical Grid

The numerical grid stretches from the inner boundary just inside the event horizon, at $\sim r_g$, to the outer boundary at $\sim 10^5 r_g$. It has a total effective resolution of $13,440 \times 4608 \times 8096$ in the disk beyond $\sim 10 r_g$. The resolution gradually drops to the base-grid resolution of $1728 \times 576 \times 1024$ at $r \sim 4 r_g$ (this increases the time step by about an order of magnitude; Section 3.3). The cell sizes at $r = (1.5, 8, 20, 50) r_g$ are $|\Delta r| \sim |\Delta \theta| \sim |\Delta \phi| \sim (0.0082, 0.011, 0.011, 0.034)$. The grid has $(0.9\text{--}1.4) \times 10^5$ blocks of size $48 \times 48 \times 64$ cells and a total of ~ 12 – 22 billion cells. The number of blocks and cells varies throughout the evolution as AMR creates and destroys blocks to focus the resolution on the disk body as it moves through the grid. For this problem, we use four levels of AMR (Section 3.3), two external (Section 3.4.1) and four internal (Section 3.4.2) layers of SMR, and five levels of LAT (Section 3.5).

Total-resolution-wise, this advanced grid improves by more than an order of magnitude on our previous simulations of 0.25–1.5 billion cells carried out on the NCSA Blue Waters supercomputer (e.g., Liska et al. 2021). Consequently, this improves the resolution of the disk to 25–30 cells, up from 7–14 cells per scale height. Typically around 4–5 scale heights of the disk reside in the highest refinement level, ensuring uniform resolution of the magnetized turbulence not only near

the midplane of the disk but also farther away from the midplane.

4.3. Performance Measurements

The speed of GRMHD codes is usually expressed in zone cycles per second. Each zone cycle represents an operation that advances a single cell in the grid for one full time step. The computational cost in terms of memory bandwidth and number of double-precision floating pointing operations is de facto constant for each zone cycle and was measured on a single GPU with NVPROF. Thus, it is trivial to calculate those metrics for our fiducial run on OLCF Summit based on the achieved zone cycles per second. Similarly, the total memory consumption can be calculated by measuring the size of a single block and multiplying this number by the total number of blocks in the grid. In addition, we calculate the MPI bandwidth by counting the number cell transfers between AMR/SMR blocks on different nodes.

The grid described in Section 4.2 is used for the largest of our weak scaling tests, which uses 5400 V100 GPUs and requires over 25 TB of GPU RAM. We use fewer blocks of the same size ($48 \times 48 \times 64$) for the smaller scaling tests. In other words, we keep the number of blocks/GPU constant by preventing H-AMR from refining beyond a certain number of blocks (this can potentially reduce the number of AMR levels). Note that since the grid for the smaller weak scaling tests becomes smaller and may have fewer AMR layers, the problem is not exactly the same, and thus these benchmarks merely serve as an indication of what performance might look like for a hypothetical simulation using a smaller grid with similar characteristics. The plotted efficiency is accurate, though, for our fiducial simulation and represents the average performance of our code, which includes the entire application performance including I/O.

We express in Section 4 our numerical efficiency as the achieved zone cycles $s^{-1} GPU^{-1}$ relative to the maximum value devised from single GPU benchmarks with a 150^3 block size (Section 4.4). Similarly, we express the weak scaling efficiency as the achieved zone cycles $s^{-1} node^{-1}$ compared to this value achieved for a single node (six V100 GPUs) using the respective grid (~ 20 blocks of size $48 \times 48 \times 64$ per GPU).

4.4. Limitations on Scaling

As long as the grid has a resolution of $\gtrsim 125^3$ per GPU, we find that the numerical efficiency remains excellent for a wide range of problems. Lowering the resolution to 50^3 per GPU without LAT or to $\sim 100^3$ per GPU with LAT gives us only $\sim 40\%$ of the GPU's maximum performance in zone cycles per second. The per-GPU resolution limitation arises because a GPU needs to keep $\sim 10^5$ CUDA threads occupied, which is not possible for smaller grids, especially when LAT reduces the total number of time steps by a factor of ~ 10 . To allow for efficient load balancing, this grid needs to be divided such that there are at least $\gtrsim 15$ blocks per GPU. Using $\lesssim 7$ blocks per GPU leads to a factor 2–3 performance decrease. Another constraint comes from the block size, which needs to exceed $\gtrsim 50^3$ cells, such that internode MPI transfers do not become prohibiting. For block sizes of $\sim 50^3$ cells we find that on average $\sim 25\%$ of the MPI interconnect bandwidth is saturated for our fiducial run (Table 1). However, since some nodes can be more MPI-intensive than others, this number can

Table 1
Time-averaged Absolute and Relative Performance of H-AMR

	Abs. Perf.	Rel. Perf.
FP64 rate	2.2 PFLOP s^{-1}	5.3%
GPU DRAM bandwidth	588 TB s^{-1}	12.1%
GPU DRAM consumption	25 TB	28%
MPI bandwidth	5.5 TB s^{-1}	24%

Note. Floating-point power (FP64), GPU DRAM bandwidth, GPU DRAM consumption, and MPI bandwidth for our fiducial run on 5400 V100 GPUs of the OLCF Summit supercomputer. This demonstrates that H-AMR does not have a “single” bottleneck resource, making efficient use of OLCF Summit.

be much higher for certain nodes and makes the MPI bandwidth the main performance bottleneck for our fiducial run. Reducing the block size to $\sim 26^3$ while keeping the total number of cells constant decreases the performance by a factor 2–3. Furthermore, the available HBM2 RAM on each 16 GB GPU can only fit a 256^3 grid.

These constraints make it imperative that every grid has to be fine-tuned to the respective system. In our case the simulation has run $\sim 10\%$ of the time on 3600 V100 GPUs, which is the minimum amount of GPUs we need owing to RAM constraints, and $\sim 90\%$ of the time on 5400 V100 GPUs, above which there are not enough blocks to effectively load-balance the grid. We find that within this range the number of zone cycles per second per GPU remains constant to within 10%.

5. Performance Results

In this section we demonstrate excellent single GPU performance, time-to-solution, scalability, and peak performance for the thin accretion disk problem presented in Section 4.

5.1. Single CPU/GPU Benchmarks

Figure 5 compares the computational performance in zone cycles per second of H-AMR on various GPU and CPU architectures for a simple test configuration without using AMR, SMR, or LAT and block size of 150^3 . On a single NVIDIA V100 GPU H-AMR attains $\sim 1, 0 \times 10^8$ zone cycles s^{-1} , which corresponds to a factor ~ 5 speedup compared to a 28-core Intel W3175X Skylake CPU (clocked at 2.8 GHz with 16 cores active), which attains 10×10^5 zone cycles s^{-1} per core using AVX-512 vectorization. Note that a zone cycle includes the predictor and corrector steps in our second-order-accurate time-stepping routine. We profiled our code using NVPROF on a single GPU and manage to saturate 25% of FP64 throughput and 57% of memory bandwidth without any register spillover on a single NVIDIA V100 GPU for a 150^3 block. We consider this a remarkable result, especially since H-AMR contains more than 10,000 lines of CUDA kernel code.

5.2. Peak Performance

From the total number of zone cycles per second achieved in our fiducial run we can easily calculate the peak performance on 5400 V100 GPUs as described in Section 4.3. The results are summarized in Table 1, which shows that H-AMR uses a significant fraction of the FP64 power, GPU memory

Table 2
Performance Metrics of H-AMR in Four Astrophysical Scenarios

	TD-4608	TD-1728	EHT-192	EHT-2304
Base resolution	$1680 \times 576 \times 1024$	$1020 \times 432 \times 288$	$304 \times 192 \times 192$	$5376 \times 2304 \times 2304$
Effective resolution	$13440 \times 4608 \times 8192$	$4080 \times 1728 \times 1152$	$304 \times 192 \times 192$	$5376 \times 2304 \times 2304$
Block size	$48 \times 46 \times 64$	$102 \times 36 \times 36$	$76 \times 32 \times 48$	$64 \times 44 \times 64$
Grid outer radius	$10^5 r_g$	$2000 r_g$	$150 r_g$	$2000 r_g$
Physical duration	$1.5 \times 10^5 r_g/c$	$10^5 r_g/c$	$10^4 r_g/c$	$10^4 r_g/c$
Hardware computational cost	3.8×10^6 GPU hr	3.3×10^4 GPU hr	18 GPU hr	1.1×10^6 GPU hr
System scale	5400 V100 GPUs	128 V100 GPUs	1 V100 GPU	6000 V100 GPUs
Number of cells	$(12-22) \times 10^9$	0.44×10^9	9.2×10^6	22×10^9
Number of real zone cycles	1.7×10^{17}	5.2×10^{15}	0.64×10^{13}	0.88×10^{17}
Number of effective zone cycles	1.5×10^{18}	1.8×10^{16}	1.6×10^{13}	4.4×10^{17}
Effective zone cycles s^{-1}	$\gtrsim 1.1 \times 10^8 \text{ GPU}^{-1}$	$\gtrsim 1.5 \times 10^8 \text{ GPU}^{-1}$	$\gtrsim 2.5 \times 10^8 \text{ GPU}^{-1}$	$\gtrsim 1.1 \times 10^8 \text{ GPU}^{-1}$
LAT \times GPU speedup	31	43	71	31
SMR speedup ($\#Time\ steps$)	3.3	1.42	1.17	6.7
AMR speedup ($\#Cells$)	35	18	1	1
AMR speedup ($\#Time\ steps$)	53	20.7	1	1
Total speedup	1.9×10^5	2.3×10^4	83	208

Note. TD-4608 is the fiducial run of a tilted thin accretion disk described in this article, which makes use of three levels of AMR. TD-1728 is an aligned thin accretion disk that makes use of three levels of AMR (Liska et al. 2022). EHT-192 is a geometrically thick accretion disk that does not use AMR (Porth et al. 2019). It represents a typical medium-resolution GRMHD simulation part of the Event Horizon Telescope Collaboration (EHTC) GRMHD simulation library. EHT-2304 is a similar simulation of a geometrically thick accretion, but performed at an extremely high resolution without AMR (Ripperda et al. 2022). The effective number of zone cycles and zone cycles s^{-1} assumes that all cells are evolved at the highest LAT level. Thus, the ratio between the number of effective and real zone cycles equals the factor by which LAT reduces the number of time steps. The LAT \times GPU speedup gives the factor by which LAT and GPUs speed up H-AMR. We compare against a 20-core Skylake CPU, on which the non-AVX-vectorized version of H-AMR has a performance of $\sim 3.5 \times 10^6$ zone cycles s^{-1} . SMR/AMR speedup ($\#Time\ steps$) gives the cost reduction factor in the number of time steps compared to a spherical grid without SMR that uses cylindrication (Tchekhovskoy et al. 2011) around the polar axis such that it is not limited by the Courant condition in the φ -direction up to a resolution of $N_\varphi = 128$. For $N_\varphi > 128$ we assume that $\Delta t^3 = \frac{128}{N_\varphi} \Delta t^2$ and calculate the time step as $\Delta t \propto 1/(1/\Delta t^1 + 1/\Delta t^2 + 1/\Delta t^3)$ (Section 2). AMR speedup ($\#Cells$) similarly gives the reduction factor in the number of cells, which is defined as the ratio between the true number of cells in the grid and the number of cells in a unigrid at the effective resolution. Multiplying all of these factors gives the total speedup.

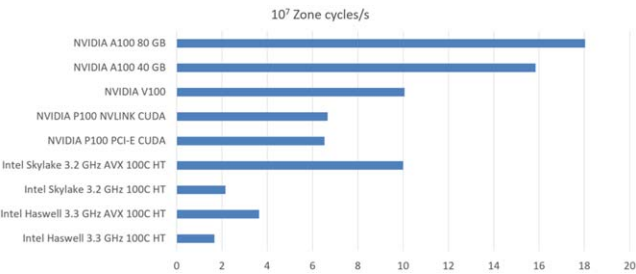


Figure 5. The computational performance of H-AMR in zone cycles per second on a 150^3 uniform grid for various generations of Intel CPUs (Haswell and Skylake) and NVIDIA GPUs (Pascal, Volta, and Ampere). HT indicates that we make use of HyperThreading on Intel CPUs. A zone cycle includes both the predictor and corrector step in our second-order-accurate time-stepping routine.

bandwidth, GPU memory size, and MPI bandwidth. This leads us to conclude that H-AMR is very well optimized across the board and makes efficient use of the resources available on OLCF Summit. Furthermore, it shows that while the MPI and GPU memory bandwidths are the current bottlenecks, implementation of new floating point heavy physics such as radiation

into H-AMR may cause the floating point capacity to become the bottleneck resource in the future.

6. Performance Summary

In this article we have presented H-AMR's performance under a scenario of extreme complexity featuring an unusually high cell count, AMR, SMR, and thousands of GPUs. In Table 2 we summarize H-AMR's performance spanning a range of complexity. These range from a typical $\sim 192^3$ GRMHD simulation (EHT-192) of a geometrically thick accretion disk that we ran on a single GPU during the GRMHD code comparison (Porth et al. 2019) to the very thin disk (TD-4608) considered in this work. We also summarize H-AMR's performance in a mid-resolution GRMHD simulation of a thin accretion disk (TD-1728) with AMR on 128 GPUs (Liska et al. 2022) and in a 2304^3 GRMHD simulation of a geometrically thick accretion disk (EHT-2304) without AMR on 6000 GPUs (Ripperda et al. 2022).

In Table 2 we calculate the speedup compared to H-AMR's performance on a 20-core Intel Skylake CPU without any advanced features (AMR, SMR, LAT, GPUs). We assume that the time step is not limited by the Courant condition in the φ -direction until a resolution of $N_\varphi = 128$ owing to cylindrication of the grid

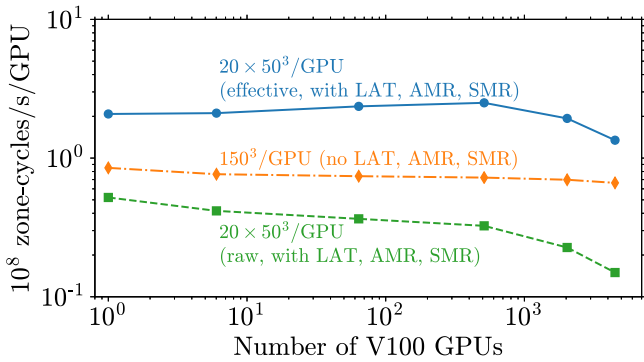


Figure 6. H-AMR shows excellent weak scaling for both simple and complex grids. For simple grids, it shows $\sim 80\%$ numerical efficiency on 900 OLCF Summit nodes = 5400 GPUs for a single block of 150^3 cells per GPU, without using AMR, SMR, or LAT (dashed-dotted orange line). For more complex grids, which use AMR and SMR, with 20 blocks of size $48 \times 48 \times 64 \sim 50^3$ cells per GPU, the efficiency at 900 nodes drops to $\sim 60\%$ (not shown for brevity). While using LAT decreases the raw parallel efficiency to $\sim 35\%$ (dashed green line), it also effectively speeds up the simulations by a factor of 4 (on one GPU) to 9 (on 5400 GPUs; solid blue line), leading to an effective numerical efficiency of nearly $\sim 200\%$ on 5400 GPUs.

(Tchekhovskoy et al. 2011). A more detailed explanation of how these speedup factors are calculated is given in the caption of Table 2. Under these assumptions, the attained speedup ranges from a factor ~ 80 for EHT-192 to a factor $\sim 1.9 \times 10^5$ for TD-4608, since TD-4608 benefits a lot from SMR and AMR. However, as the performance of EHT-2304 demonstrates, even in the absence of any AMR, the speedup can exceed a factor $\gtrsim 200$ at higher resolution where squeezing of cells near the pole makes the simulation significantly more expensive without SMR (Section 3.4).

6.1. Scalability

The simulation presented in Section 4 scales up to 900 OLCF Summit nodes and 5400 V100 GPUs. Each GPU on average contains 20 blocks, each of $48 \times 48 \times 64 \sim 50^3$ cells. The green dashed line in Figure 6 shows that with the use of SMR, AMR, and LAT we obtain what might seem like a disappointing numerical efficiency (with the baseline set by the number of zone cycles per second on a single GPU for a 150^3 block) of $\sim 18\%$ and a weak scaling efficiency (using a single six-GPU node as a baseline with a grid containing 120 blocks of size $48 \times 48 \times 64$) of $\sim 35\%$ at 900 Summit nodes. One reason for the seemingly low parallel scaling efficiency is the difficulty of load balancing that arrives with LAT: without LAT, the weak scaling efficiency increases to a respectable $\sim 60\%$. However, by decreasing the number of time steps, LAT speeds up the simulations by factors of ~ 4 (on a single V100 GPU) to ~ 9 (on 5400 V100 GPUs): this effectively brings the parallel numerical efficiency to $\sim 200\%$ at 900 Summit nodes = 5400 V100 GPUs, as shown by the solid blue line in Figure 6. This corresponds to (effectively) $\sim 1.175 \times 10^8$ zone cycles s^{-1} GPU $^{-1}$. Thus, the net result of LAT is a substantial increase in effective performance and parallel scaling efficiency.

The other reason for lower numerical efficiency in our fiducial problem compared to simpler test problems is the smaller block size. For illustration, consider an idealized weak scaling test, with a much larger block size of 150^3 cells per GPU; we disable SMR, AMR, and LAT for this test. The orange dashed-dotted line in Figure 6 shows that this leads to an excellent parallel weak scaling efficiency of $\sim 80\%$ at 900

Summit nodes, primarily because inefficiencies in load balancing related to LAT go away (Section 3.5), and the MPI interconnect bandwidth is not anymore the main bottleneck owing to the larger block size and therefore smaller fraction of boundary cells. Note that we would not be able to use such large blocks of 150^3 cells for our fiducial problem, as there would be an insufficient number of blocks per GPU to effectively load-balance the grid and the blocks would be too big to properly focus on the very thin $h/r = 0.02$ disk.

7. Code Validation

For the purpose of code development and validation, H-AMR maintains an independently developed and maintained OpenMP and MPI-parallelized C version running on CPUs. More specifically, we usually develop new features in the C version and, after extensive testing, port them to the CUDA/OpenCL version. Subsequently, we verify that the output of both versions agrees up to machine precision for various scenarios. We refer the reader to Porth et al. (2019) for code validation of H-AMR against its peer codes, performed in the context of a community-wide GRMHD code comparison project, and to the Appendix of Liska et al. (2018) for validation of H-AMR for tilted accretion flows evolved on a spherical grid. We supplement these tests in this article with a suite of linear and nonlinear wave tests and two off-center spherical explosions.

7.1. Linear Wave Tests

To verify second-order convergence of the boundary conditions between coarse and fine layers in H-AMR, we have performed a variety of 3D dimensional linear wave tests (with an $O(10^{-5})$ amplitude), which include the fast, slow, Alfvén, and entropy waves (e.g., Gammie et al. 2003). We rotate the wavevector by 45° with respect to the x -, y -, and z -axes on a Cartesian grid and force these waves to propagate through a grid refinement boundary, which also features a jump in time stepping (top panels of Figure 7). We evolve each respective wave for two periods and then compute the L^1 error, which is defined as the absolute value of the finite difference between the initial and final states for the primitive variables. In the bottom panel of Figure 7 we demonstrate that H-AMR, as expected, converges at second order for all waves.

7.2. Nonlinear Wave Tests

To verify the ability of H-AMR to capture shock waves under challenging conditions, we perform a series of relativistic shock tube tests as described in Komissarov (1999). We evolve the shock tubes on a 1D Cartesian grid in Minkowskian space with no refinement and consider both a MINMOD and PPM slope limiter (without contact steepener or shock flattener). We use a Courant factor $C_f = 0.8$ except for the fast shock, for which we use $C_f = 0.2$. We compare in Figure 8 an extremely high resolution shock tube ($\Delta x = 1.1 \times 10^{-3}$) to a lower-resolution shock tube ($\Delta x = 2.2 \times 10^{-2}$). The shock fronts in the low-resolution shock tube are, as expected, less resolved. While the very diffusive MINMOD limiter does not produce spurious oscillations near shock fronts, some relatively small spurious oscillations are detectable with the less diffusive PPM limiter. This is a well-known disadvantage of PPM limiters that can be addressed with an optional shock flattener implemented in H-AMR, which reduces the order of reconstruction near

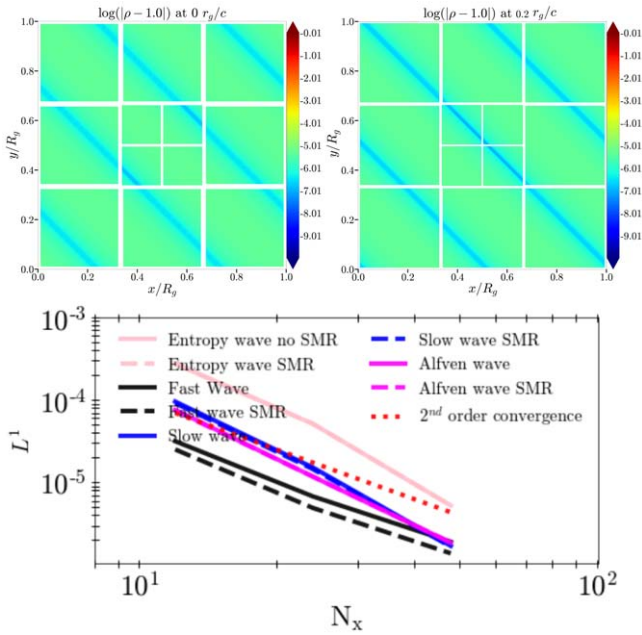


Figure 7. Top panels: a density isocontour rendering at two different times for a fast wave. Block boundaries are illustrated using white lines. For our wave tests we start with a grid that has $3 \times 3 \times 3$ blocks and refine the innermost block. Bottom panel: a plot of the L1 error as a function of the linear resolution N_x . H-AMR achieves second-order convergence for all waves irrespective of refinement boundaries.

shock fronts (Colella & Woodward 1984). These tests verify the robustness and stability of the shock-capturing scheme in H-AMR when handling shocks in relativistic plasma.

7.3. Off-center Spherical Explosion

To quantify numerical errors associated with gas passing through the polar singularity and internal/external SMR layers, we perform an off-center explosion that propagates through the poles. During this test, we use one layer of external SMR, two layers of internal SMR, and four levels of LAT. The test is performed on a logarithmically spaced spherical grid instead of a Cartesian grid to also validate the correct implementation of the geometric source terms. The total resolution is $300 \times 240 \times 240$. The initial pressure and density are set to a constant $p = 3 \times 10^{-5}$ and $\rho = 10^{-4}$ within a sphere of radius $r_s = 2r_g$ that is centered at $(x_0, y_0, z_0) = (3, 0, 10)r_g$. We use an exponential cutoff to make the transition between the sphere and ambient medium smoother. In addition to the hydrodynamic explosion, we perform a similar test where the explosion is threaded by a magnetic field described by vector potential $A_\phi \propto (r_s - \sqrt{(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2})$ for $r < r_s$ and $A_\phi = 0$ for $r > r_s$. The magnetic field is subsequently normalized such that $\beta^{\max} = p_g^{\max} / p_b^{\max} = 10$. Here p_g^{\max} and p_b^{\max} are the maximum gas and magnetic pressure in the grid, respectively.

The results of these off-center explosions are illustrated in the left (hydrodynamic explosion) and right (magnetized explosion) panels of Figure 9. Both explosions manage to maintain their circular shape when passing through the polar axis. The minor distortions around the polar singularity are similar to those observed in other GRMHD codes that use transmissive boundary conditions (e.g. Mewes et al. 2018, 2020). This demonstrates that H-AMR is capable of accurately

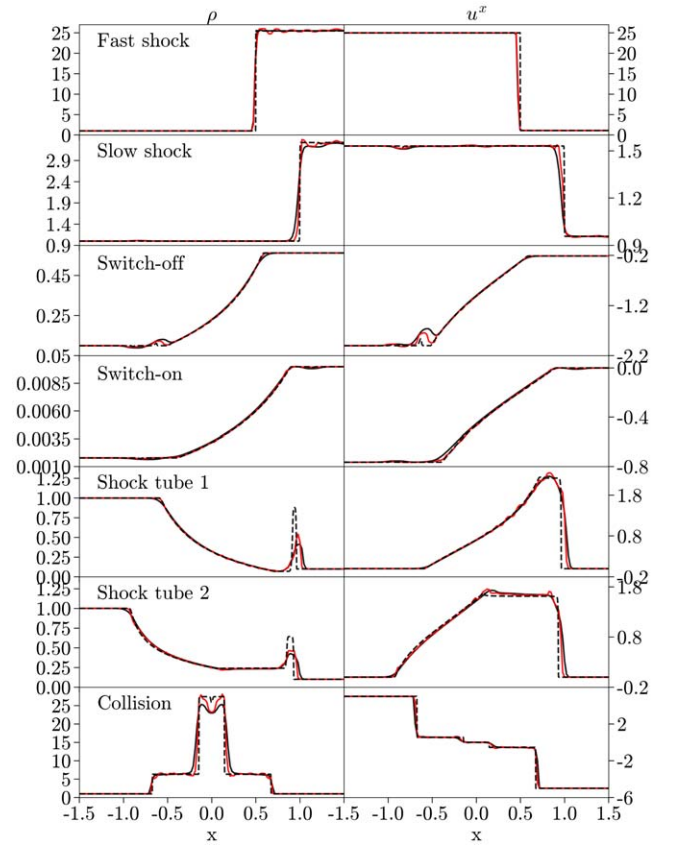


Figure 8. Relativistic shock tubes taken from Komissarov (1999) with density (ρ) in the left panels and velocity (u^x) in the right panels. Reconstruction with the MINMOD limiter is shown in black and with the PPM limiter in red. The low-resolution shock tubes with $\Delta x = 2.2 \times 10^{-2}$ (solid lines) are less sharp than the high-resolution shock tubes with $\Delta x = 1.1 \times 10^{-3}$ (dashed lines); however, they converge to the same solution. This verifies the ability of H-AMR to handle shock waves in relativistic plasma.

evolving plasma around the pole and can even maintain some form of convergence when the gas passes directly through the polar singularity.

To avoid repeated passage of, for example, a kink-unstable jet through the polar singularity, H-AMR has the ability to “rotate” the setup with respect to the spherical grid such that the jet propagates along the equatorial plane instead (e.g., Gottlieb et al. 2022a, 2022b, 2022c). The physical equivalence between rotated and nonrotated setups was demonstrated in the Appendix of Liska et al. (2018).

8. Discussion

8.1. Advancements in Computational Methods

The extensive optimizations presented in this paper bring state-of-the-art GRMHD simulations to the exascale level, e.g., to clusters with more than 10^{18} FLOP s^{-1} of FP64 performance, using a highly efficient grid capable of tackling problems with many orders of magnitude in scale separation. We carried out scaling tests up to 5400 NVIDIA V100 GPUs on OLCF Summit, culminating in our fiducial simulation: the largest GRMHD simulation to date that utilizes four AMR levels, with the block size of $48 \times 64 \times 64$ cells, and achieves an astonishingly high effective resolution in the disk of $13,440 \times 4608 \times 8096$ cells. The simulation cost is around 4

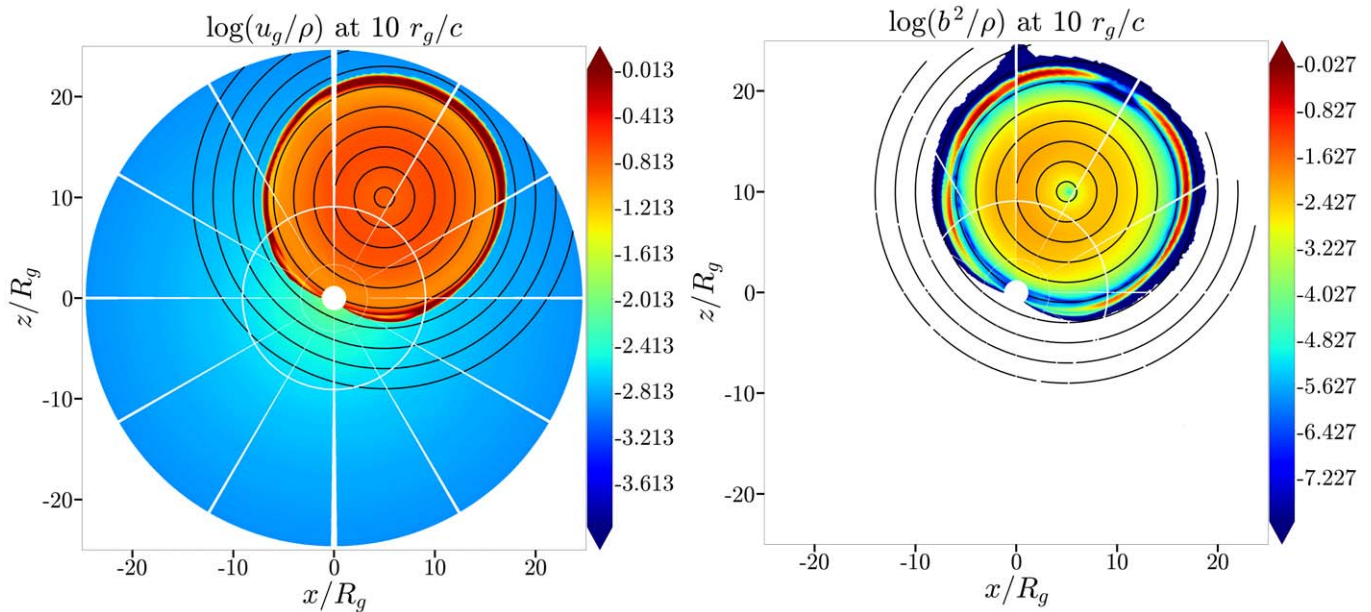


Figure 9. Left panel: the ratio between internal energy (u_g) and rest mass density (ρ) during a spherical off-center explosion that passes through two internal and one external SMR layer in addition to the polar axis. The different meshblocks are separated by white lines. Black lines illustrate the distance from the center of the explosion in steps of $1r_g$. The front of the shock wave becomes less resolved at larger radii owing to the logarithmic spacing of the grid. Right panel: the ratio between the magnetic energy (b^2) and rest mass density (ρ) during a magnetized off-center explosion that passes through two internal and one external SMR layer in addition to the polar axis. When the explosion passes through the polar singularity, only minor artifacts in the internal energy and magnetic field strength are detected, which is consistent with other (GR)MHD codes that utilize similar (transmissive) boundary conditions (e.g., Mewes et al. 2018, 2020).

million NVIDIA V100 GPU hr, which corresponds to roughly 800 million Skylake CPU core hours (Section 4.4).

H-AMR features a set of optimizations that collectively boost the performance of GRMHD simulations by at least 2 orders of magnitude for simple problems that do not benefit from an adaptive mesh. This increases to 5 orders of magnitude for problems requiring adaptive meshes and very high resolutions in the polar region (see Section 4.2). This speedup is attained by the following means. An NVIDIA V100 GPU typically provides a factor of ~ 40 speedup compared to a non-AVX-vectorized version of H-AMR running on 16 Intel Skylake cores (AVX vectorization improves the performance by a factor ~ 3). Large gains also come from the advanced treatment of the polar region during SMR/AMR, which provides a typical factor 5–20 reduction in the number of time steps, depending on the resolution. LAT reduces the number of time steps by another factor of ~ 3 –10 depending on the resolution of the grid. Finally, AMR can reduce the number of cells in the grid by a factor of ~ 10 –100 for many complex problems such as thin accretion disks and large-scale jets.

This paves the way for a next generation of GRMHD simulations with unprecedented resolutions and record-breaking simulated timescales, substantially advancing the understanding of accreting black holes. For example, H-AMR made it possible to simulate even the most challenging accretion disk configurations naturally expected in nature (i.e., thin and tilted) over timescales that are relevant to astronomical observations (Liska et al. 2019a, 2019b, 2021). Liska et al. (2019a) demonstrated, for the first time in GRMHD, that tilted thin magnetized disks align with the black hole midplane as theoretically predicted (Bardeen & Petterson 1975), solving a 40 yr old problem in astrophysical disk dynamics. Further, not only this article but also Liska et al. (2021) and Musoke et al. (2022) simulated disk tearing in highly tilted thin disks (Figure 3), potentially explaining some types of QPOs

observed in XRBs (e.g., Kalamkar et al. 2016), as well as holding important consequences for galactic evolution (e.g., King et al. 2005). Using H-AMR, Chatterjee et al. (2019) simulated the longest extent (in both temporal and spatial dimensions) relativistic jets from an accreting black hole in 2D GRMHD, making it possible to follow the jet over a range of $\gtrsim 5$ orders of magnitude in distance and enable the study of the jet’s deceleration process as it carves out its path through the environment. Following this line of study, H-AMR would finally make it possible to simulate black-hole-launched jets and connect distances from the event horizon up to extragalactic distances, closing the gap between GRMHD and cosmological simulations of galaxy evolution.

H-AMR also played an important role in interpreting (e.g., Porth et al. 2019; Chatterjee et al. 2020; Ripperda et al. 2022; Event Horizon Telescope Collaboration et al. 2022a) the black hole images from Event Horizon Telescope (EHT) observations of the supermassive black holes M87* (e.g., Event Horizon Telescope Collaboration et al. 2019a, 2019b, 2021) and Sagittarius A* (e.g., Event Horizon Telescope Collaboration et al. 2022b, 2022c). In particular, H-AMR enabled the first study of how general relativity warps jets (and their accretion disks) that are misaligned with the black hole spin axis, ultimately affecting the near-event horizon image of a black hole (Liska et al. 2019a; Chatterjee et al. 2020). The ability to carry out the simulations at extremely high resolutions enables H-AMR to attain in global simulations (Ripperda et al. 2022) the resolutions typically associated with local simulations. Such global modeling has already enabled a significant improvement in our understanding of how black holes consume and eject gas. It has been thought that the formation of relativistic jets by an accreting black hole requires the presence of large-scale vertical magnetic flux to begin with, and that absent such a field—e.g., if only a purely toroidal magnetic field is present—the simulations do not produce jets

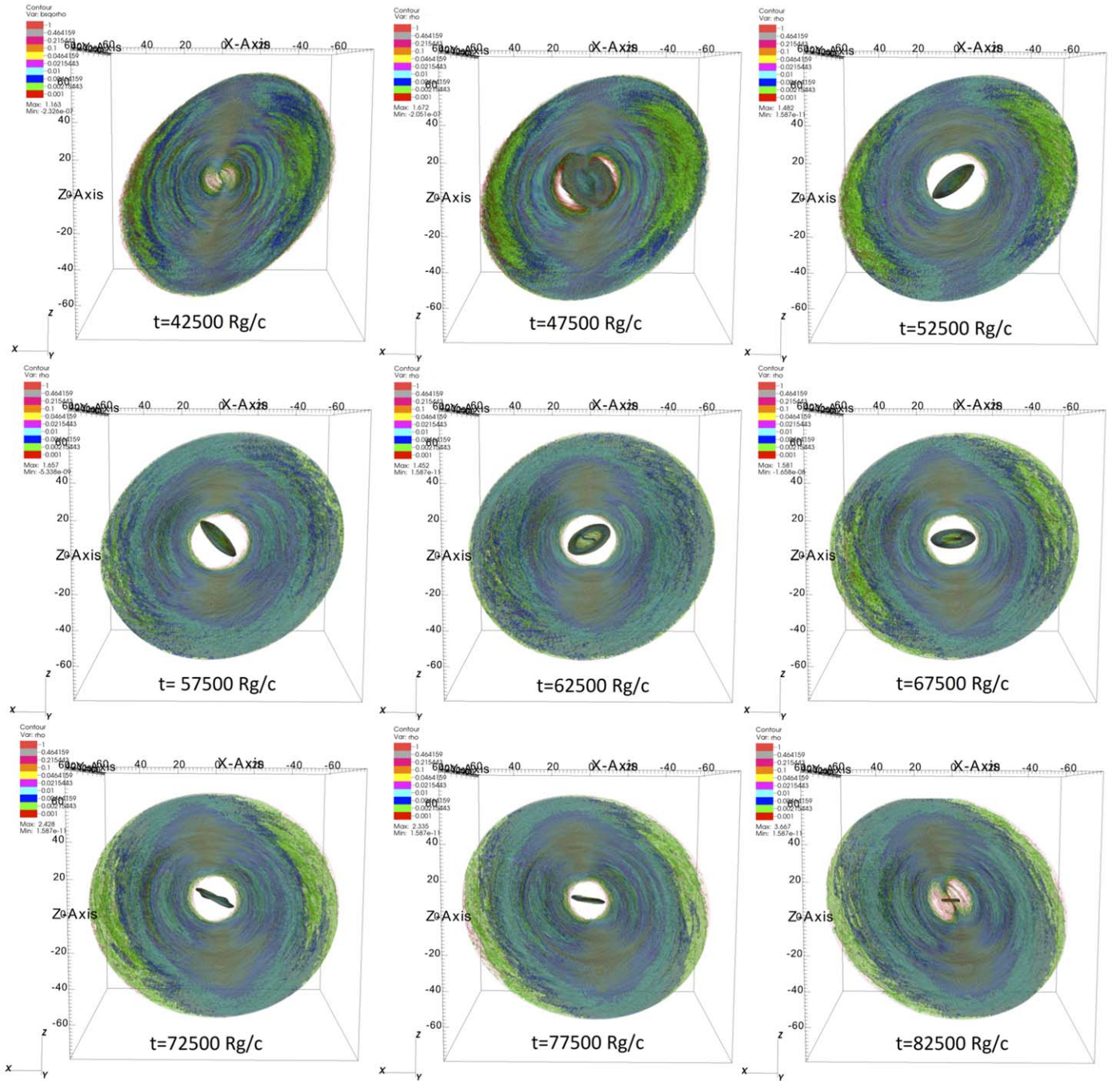


Figure 10. A 3D isocontour rendering of density at different times in our simulation, showing a single tearing cycle of the disk: at the initial time shown, $t = 42,500 r_g/c$ (top left panel), the disk is intact. However, at $t = 47,500 r_g/c$, the inner disk of size $\sim 20 r_g$ tears off from the outer disk and starts to precess independently of the outer disk. Eventually, the inner precessing disk gets consumed by the black hole, and the entire disk appears intact again at the last time shown, $t = 82,500 r_g/c$ (bottom right panel). During each tearing cycle, the torque from the spinning black hole overcomes the viscous torques holding the disk together, and the disk tears apart. This results in a rapidly precessing inner subdisk physically detached from the outer disk. This subdisk can sustain several precession periods before it shrinks and disappears into the black hole and the tearing cycle repeats. Disk tearing and precession is an attractive model to explain type C QPOs in the light curves of XRBs (e.g., van der Klis 2006).

(Beckwith et al. 2008a; McKinney et al. 2012). However, our simulations have revealed that such configurations are actually capable of generating large-scale vertical magnetic flux and forming relativistic jets (Liska et al. 2020).

Summing up, H-AMR is in a prime position to address important questions about the dynamic nature of accretion disks and jets such as the hitherto-unexplained trigger behind the observed change in disk geometry from thick to thin (e.g.,

Belloni 2010; Liska et al. 2022), the evolution of jet morphologies (known as the FR I/II dichotomy; Fanaroff & Riley 1974), and the origins of particle acceleration in both disks and jets (Sironi et al. 2015; Ripperda et al. 2022). The high speed of H-AMR makes it attractive to account for departures from ideal MHD physics such as radiation (e.g., Sadowski et al. 2013; Ryan et al. 2017; Foucart 2018) and synchrotron cooling of electrons (e.g., Fragile & Meier 2009;

Dibi et al. 2012; Drappeau et al. 2013; Yoon et al. 2020), resistivity (e.g., Ripperda et al. 2019a, 2019b; Mignone et al. 2019), and two-temperature thermodynamics (e.g., Ressler et al. 2015; Chael et al. 2017; Ryan et al. 2018; Liska et al. 2022), which may lead to important insights into many of the abovementioned problems. Currently, H-AMR's speed is only (MPI bandwidth) limited by waiting times for internode communication, which leaves time for extra computations. Therefore, more complex physics can be implemented (almost) free of charge. Such numerical experiments are expected to scale well on exascale systems, even if those systems feature slower internode communication than in the simulations presented here and conducted at OLCF Summit.





8.2. Scientific Results

For the scaling tests presented in this article we have used a real-world problem involving a highly tilted (by 65°) very thin ($h/r=0.02$) accretion disk threaded by a strong toroidal magnetic field ($\beta\sim 10$) around a rapidly spinning black hole ($a\sim 0.94$). These simulations are more extensively described in Musoke et al. (2022). Interestingly, unlike thick disks threaded with toroidal magnetic field that tend to develop large-scale poloidal magnetic flux through a dynamo process and associated relativistic jets (Liska et al. 2020), our thin disk did not produce any jets, as seen in Figure 3. In fact, due to the complete absence of jets, the outcome resembles the high/soft state in XRBs, and as such this is the first GRMHD simulation to reach this exceptionally challenging jet-less regime, in the complete absence of initial poloidal magnetic flux. While in previous work we needed to seed the accretion disk with a rather strong poloidal magnetic field as is only expected in the intermediate states (Liska et al. 2019a, 2019b, 2021), the powerful NVIDIA V100 GPUs of OLCF Summit allowed us to resolve the weak toroidal magnetic field with $\gtrsim 20$ cells per MRI wavelength and the disk scale height with 25–30 cells.

As illustrated in Figures 3 and 10 and in the accompanying movie,⁶ we find that tilted thin accretion disks, initially threaded with a purely toroidal magnetic field (i.e., containing no poloidal magnetic flux to begin with), can tear apart into multiple precessing subdisks. We also find that, even in the absence of a Blandford & Znajek (1977) jet, the inner regions of a magnetized accretion disk can sporadically align with the black hole midplane, as predicted more than four decades ago by Bardeen & Petterson (1975) and seen in such disks for the first time (see Figure 3). The movie⁶ of the simulation shows multiple ($\gtrsim 4$) cycles of such tearing events that result in the formation of a precessing inner subdisk of about the same size. The emission from such precessing subdisks is expected to be quasiperiodic. Thus, the tearing of tilted disks is an attractive mechanism for producing coherent QPOs, now testable via GRMHD simulations for the first time (e.g., Musoke et al. 2022). Improved understanding of the evolution of QPO frequency and amplitude during XRB outbursts may lead to unique constraints on black hole spin magnitude and/or disk geometry (e.g., Ingram et al. 2009; Motta et al. 2015) and provide new insights into the physics driving XRB spectral state transitions (e.g., Ferreira et al. 2006; Begelman & Armitage 2014; Marcel et al. 2018a, 2018b; Liska et al. 2019a, 2020).

An award of computer time was provided by the Innovative and Novel Computational Impact on Theory and Experiment (INCITE) and the ASCR Leadership Computing Challenge (ALCC) programs under award PHY129. This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under contract DE-AC05-00OR22725. We acknowledge PRACE for awarding us access to JUWELS Booster at GCS@JSC, Germany. M.L. was supported by the John Harvard Distinguished Science and ITC Fellowships and the 21-ATP21-0077 NASA ATP award. M.K. was supported by the Netherlands Organisation for Scientific Research (NWO) Spinoza Prize, C.H. by the Amsterdam Science Talent Scholarship, A.I. by a Royal Society University Research Fellowship, A.T. by the National Science Foundation grants AST-2206471, AST-2009884, AST-2107839, AST-1815304, OAC-2031997, and AST-1911080, and S.M., K.C., and D.Y. by the NWO VICI grant (No. 639.043.513).

ORCID iDs

N. Kaaz  <https://orcid.org/0000-0002-5375-8232>
 S. Markoff  <https://orcid.org/0000-0001-9564-0876>
 A. Ingram  <https://orcid.org/0000-0002-5311-9078>
 M. van der Klis  <https://orcid.org/0000-0003-0070-9872>

References

- Andalman, Z. L., Liska, M. T. P., Tchekhovskoy, A., Coughlin, E. R., & Stone, N. 2022, *MNRAS*, **510**, 1627
- Antón, L., Zanotti, O., Miralles, J. A., et al. 2006, *ApJ*, **637**, 296
- Balbus, S. A., & Hawley, J. F. 1991, *ApJ*, **376**, 214
- Balsara, D. S. 2001, *JCoPh*, **174**, 614
- Bardeen, J. M., & Petterson, J. A. 1975, *ApJL*, **195**, L65
- Beckwith, K., Hawley, J. F., & Krolik, J. H. 2008a, *ApJ*, **678**, 1180
- Beckwith, K., Hawley, J. F., & Krolik, J. H. 2008b, *MNRAS*, **390**, 21
- Begelman, M. C., & Armitage, P. J. 2014, *ApJL*, **782**, L18
- Bégué, D., Pe'er, A., Zhang, G., Zhang, B., & Pevzner, B. 2022, arXiv:2205.02484
- Belloni, T. M. 2010, in *States and Transitions in Black Hole Binaries*, ed. T. Belloni (Berlin: Springer), 53
- Berger, M. J., & Colella, P. 1989, *JCoPh*, **82**, 64
- Berger, M. J., & Oliner, J. 1984, *JCoPh*, **53**, 484
- Blandford, R. D., & Znajek, R. L. 1977, *MNRAS*, **179**, 433
- Caproni, A., Abraham, Z., Livio, M., & Mosquera Cuesta, H. J. 2007, *MNRAS*, **379**, 135
- Caproni, A., Abraham, Z., & Mosquera Cuesta, H. J. 2006, *ApJ*, **638**, 120
- Carter Edwards, H., Trott, C. R., & Sunderland, D. 2014, *JPDC*, **74**, 3202
- Chael, A. A., Narayan, R., & Sadowski, A. 2017, *MNRAS*, **470**, 2367
- Chatterjee, K., Liska, M., Tchekhovskoy, A., & Markoff, S. B. 2019, *MNRAS*, **490**, 2200
- Chatterjee, K., Younsi, Z., Liska, M., et al. 2020, *MNRAS*, **499**, 362
- Colella, P., & Woodward, P. R. 1984, *JCoPh*, **54**, 174
- Courant, R., Friedrichs, K., & Lewy, H. 1928, *MatAn*, **100**, 32
- Curd, B., & Narayan, R. 2019, *MNRAS*, **483**, 565
- Davelaar, J., Olivares, H., Porth, O., et al. 2019, *A&A*, **632**, A2
- De Villiers, J.-P., Hawley, J. F., & Krolik, J. H. 2003, *ApJ*, **599**, 1238
- Dibi, S., Drappeau, S., Fragile, P. C., Markoff, S., & Dexter, J. 2012, *MNRAS*, **426**, 1928
- Drappeau, S., Dibi, S., Dexter, J., Markoff, S., & Fragile, P. C. 2013, *MNRAS*, **431**, 2872
- Evans, C. R., & Hawley, J. F. 1988, *ApJ*, **332**, 659
- Event Horizon Telescope Collaboration, Akiyama, K., Alberdi, A., et al. 2019a, *ApJL*, **875**, L1
- Event Horizon Telescope Collaboration, Akiyama, K., Alberdi, A., et al. 2019b, *ApJL*, **875**, L5
- Event Horizon Telescope Collaboration, Akiyama, K., Alberdi, A., et al. 2022a, *ApJL*, **930**, L16
- Event Horizon Telescope Collaboration, Akiyama, K., Alberdi, A., et al. 2022b, *ApJL*, **930**, L12

⁶ www.youtube.com/watch?v=rI0jKufzcvI

- Event Horizon Telescope Collaboration, Akiyama, K., Alberdi, A., et al. 2022c, *ApJL*, **930**, L15
- Event Horizon Telescope Collaboration, Akiyama, K., Algaba, J. C., et al. 2021, *ApJL*, **910**, L13
- Fanaroff, B. L., & Riley, J. M. 1974, *MNRAS*, **167**, 31P
- Ferreira, J., Petrucci, P.-O., Henri, G., Saugé, L., & Pelletier, G. 2006, *A&A*, **447**, 813
- Foucart, F. 2018, *MNRAS*, **475**, 4186
- Fragile, P. C., & Anninos, P. 2005, *ApJ*, **623**, 347
- Fragile, P. C., Blaes, O. M., Anninos, P., & Salmonson, J. D. 2007, *ApJ*, **668**, 417
- Fragile, P. C., & Meier, D. L. 2009, *ApJ*, **693**, 771
- Gammie, C. F., McKinney, J. C., & Tóth, G. 2003, *ApJ*, **589**, 444
- Gardiner, T. A., & Stone, J. M. 2005, *JCoPh*, **205**, 509
- Gottlieb, O., Lalakos, A., Bromberg, O., Liska, M., & Tchekhovskoy, A. 2022a, *MNRAS*, **510**, 4962
- Gottlieb, O., Liska, M., Tchekhovskoy, A., et al. 2022b, *ApJL*, **933**, L9
- Gottlieb, O., Moseley, S., Ramirez-Aguilar, T., et al. 2022c, *ApJL*, **933**, L2
- Greene, J., Bailyn, C. D., & Orosz, J. A. 2001, *ApJ*, **554**, 1290
- Grete, P., Glines, F. W., & O'Shea, B. W. 2021, *ITPDS*, **32**, 85
- Hamlin, N. D., & Newman, W. I. 2013, *PhRvE*, **87**, 043101
- Harten, A., Lax, P. D., & Leer, B. v. 1983, *SIAMR*, **25**, 35
- Hawley, J. F., & Krolik, J. H. 2006, *ApJ*, **641**, 103
- Hirose, S., Krolik, J. H., De Villiers, J.-P., & Hawley, J. F. 2004, *ApJ*, **606**, 1083
- Hjellming, R. M., & Rupen, M. P. 1995, *Natur*, **375**, 464
- Ingram, A., Done, C., & Fragile, P. C. 2009, *MNRAS*, **397**, L101
- Ingram, A., van der Klis, M., Middleton, M., et al. 2016, *MNRAS*, **461**, 1967
- Ivanov, P. B., & Illarionov, A. F. 1997, *MNRAS*, **285**, 394
- Jones, M. L., Hickox, R. C., Black, C. S., et al. 2016, *ApJ*, **826**, 12
- Kalamkar, M., Casella, P., Uttley, P., et al. 2016, *MNRAS*, **460**, 3284
- King, A. R., Lubow, S. H., Ogilvie, G. I., & Pringle, J. E. 2005, *MNRAS*, **363**, 49
- Komissarov, S. S. 1999, *MNRAS*, **308**, 1069
- Lalakos, A., Gottlieb, O., Kaaz, N., et al. 2022, *ApJL*, **936**, L5
- LeVeque, R. J. 2002, *Finite Volume Methods for Hyperbolic Problems* (Cambridge: Cambridge Univ. Press)
- Levermore, C. D. 1984, *JQSRT*, **31**, 149
- Liska, M., Hesp, C., Tchekhovskoy, A., et al. 2018, *MNRAS*, **474**, L81
- Liska, M., Hesp, C., Tchekhovskoy, A., et al. 2019b, arXiv:1901.05970
- Liska, M., Hesp, C., Tchekhovskoy, A., et al. 2021, *MNRAS*, **507**, 983
- Liska, M., Tchekhovskoy, A., Ingram, A., & van der Klis, M. 2019a, *MNRAS*, **487**, 550
- Liska, M., Tchekhovskoy, A., & Quataert, E. 2020, *MNRAS*, **494**, 3656
- Liska, M. T. P., Musoke, G., Tchekhovskoy, A., Porth, O., & Beloborodov, A. M. 2022, *ApJL*, **935**, L1
- Lodato, G., & Price, D. J. 2010, *MNRAS*, **405**, 1212
- Lubow, S. H., Ogilvie, G. I., & Pringle, J. E. 2002, *MNRAS*, **337**, 706
- Marcel, G., Ferreira, J., Petrucci, P. O., et al. 2018a, *A&A*, **615**, A57
- Marcel, G., Ferreira, J., Petrucci, P. O., et al. 2018b, *A&A*, **617**, A46
- McKinney, J. C. 2005, *ApJL*, **630**, L5
- McKinney, J. C. 2006, *MNRAS*, **368**, 1561
- McKinney, J. C., & Gammie, C. F. 2004, *ApJ*, **611**, 977
- McKinney, J. C., Tchekhovskoy, A., & Blandford, R. D. 2012, *MNRAS*, **423**, 3083
- McKinney, J. C., Tchekhovskoy, A., & Blandford, R. D. 2013, *Sci*, **339**, 49
- Mewes, V., Zlochower, Y., Campanelli, M., et al. 2018, *PhRvD*, **97**, 084059
- Mewes, V., Zlochower, Y., Campanelli, M., et al. 2020, *PhRvD*, **101**, 104007
- Mignone, A. 2014, *JCoPh*, **270**, 784
- Mignone, A., & Bodo, G. 2005, *MNRAS*, **364**, 126
- Mignone, A., & Bodo, G. 2006, *MNRAS*, **368**, 1040
- Mignone, A., Mattia, G., Bodo, G., & Del'Zanna, L. 2019, *MNRAS*, **486**, 4252
- Mignone, A., Ugliano, M., & Bodo, G. 2009, *MNRAS*, **393**, 1141
- Miller-Jones, J. C. A., Tetarenko, A. J., Sivakoff, G. R., et al. 2019, *Natur*, **569**, 374
- Misner, C. W., Thorne, K. S., & Wheeler, J. A. 1973, *Gravitation* (Princeton, NJ: Princeton Univ. Press)
- Motta, S. E., Casella, P., Henze, M., et al. 2015, *MNRAS*, **447**, 2059
- Musoke, G., Liska, M., Porth, O., van der Klis, M., & Ingram, A. 2022, arXiv:2201.03085
- Narayan, R., & Yi, I. 1994, *ApJL*, **428**, L13
- Narayan, R., & Yi, I. 1995, *ApJ*, **444**, 231
- Nealon, R., Price, D. J., & Nixon, C. J. 2015, *MNRAS*, **448**, 1526
- Newman, W. I., & Hamlin, N. D. 2014, *SJSC*, **36**, B661
- Nixon, C., King, A., Price, D., & Frank, J. 2012, *ApJL*, **757**, L24
- Noble, S. C., Gammie, C. F., McKinney, J. C., & Del Zanna, L. 2006, *ApJ*, **641**, 626
- Noble, S. C., Krolik, J. H., & Hawley, J. F. 2009, *ApJ*, **692**, 411
- Noble, S. C., Krolik, J. H., & Hawley, J. F. 2010, *ApJ*, **711**, 959
- Ogilvie, G. I. 1999, *MNRAS*, **304**, 557
- Papaloizou, J. C. B., & Pringle, J. E. 1983, *MNRAS*, **202**, 1181
- Penna, R. F., McKinney, J. C., Narayan, R., et al. 2010, *MNRAS*, **408**, 752
- Pons, J. A., Font, J. A., Ibanez, J. M., Martí, J. M., & Miralles, J. A. 1998, *A&A*, **339**, 638
- Porth, O., Chatterjee, K., Narayan, R., et al. 2019, *ApJS*, **243**, 26
- Porth, O., Olivares, H., Mizuno, Y., et al. 2017, *ComAC*, **4**, 1
- Ressler, S. M., Tchekhovskoy, A., Quataert, E., Chandra, M., & Gammie, C. F. 2015, *MNRAS*, **454**, 1848
- Ressler, S. M., Tchekhovskoy, A., Quataert, E., & Gammie, C. F. 2017, *MNRAS*, **467**, 3604
- Rezzolla, L., & Zanotti, O. 2013, in *Relativistic Hydrodynamics*, ed. L. Rezzolla & O. Zanotti (Oxford: Oxford Univ. Press)
- Ripperda, B., Bacchini, F., Porth, O., et al. 2019a, *ApJS*, **244**, 10
- Ripperda, B., Liska, M., Chatterjee, K., et al. 2022, *ApJL*, **924**, L32
- Ripperda, B., Porth, O., Sironi, L., & Keppens, R. 2019b, *MNRAS*, **485**, 299
- Ryan, B. R., Gammie, C. F., Fromang, S., & Kestener, P. 2017, *ApJ*, **840**, 6
- Ryan, B. R., Ressler, S. M., Dolence, J. C., Gammie, C., & Quataert, E. 2018, *ApJ*, **864**, 126
- Sądowski, A., Narayan, R., Tchekhovskoy, A., & Zhu, Y. 2013, *MNRAS*, **429**, 3533
- Sądowski, A., Wielgus, M., Narayan, R., et al. 2016, *MNRAS*, **466**, 705
- Schive, H.-Y., Zuhone, J. A., Goldbaum, N. J., et al. 2018, *MNRAS*, **481**, 4815
- Shafee, R., McKinney, J. C., Narayan, R., et al. 2008, *ApJL*, **687**, L25
- Shakura, N. I., & Sunyaev, R. A. 1973, *A&A*, **24**, 337
- Shiokawa, H., Krolik, J. H., Cheng, R. M., Piran, T., & Noble, S. C. 2015, *ApJ*, **804**, 85
- Sikora, M., Łukasz, S., & Lasota, J.-P. 2007, *ApJ*, **658**, 815
- Sironi, L., Petropoulou, M., & Giannios, D. 2015, *MNRAS*, **450**, 183
- Skinner, M. A., Dolence, J. C., Burrows, A., Radice, D., & Vartanyan, D. 2019, *ApJS*, **241**, 7
- Stella, L., & Vietri, M. 1998, *ApJL*, **492**, L59
- Stone, J. M., & Gardiner, T. 2009, *NewA*, **14**, 139
- Stone, J. M., Tomida, K., White, C. J., & Felker, K. G. 2020, *ApJS*, **249**, 4
- Tchekhovskoy, A. 2019, HARMPI: 3D massively parallel general relativistic MHD code, Astrophysics Source Code Library, ascl:1912.014
- Tchekhovskoy, A., McKinney, J. C., & Narayan, R. 2007, *MNRAS*, **379**, 469
- Tchekhovskoy, A., Narayan, R., & McKinney, J. C. 2011, *MNRAS*, **418**, L79
- van der Klis, M. 2006, in *Compact Stellar X-ray Sources*, ed. W. Lewin & M. van der Klis (Cambridge: Cambridge Univ. Press), 39
- van der Klis, M., Jansen, F., van Paradijs, J., et al. 1985, *Natur*, **316**, 225
- Volonteri, M., Madau, P., Quataert, E., & Rees, M. J. 2005, *ApJ*, **620**, 69
- White, C. J., Stone, J. M., & Gammie, C. F. 2016, *ApJS*, **225**, 22
- Yoon, D., Chatterjee, K., Markoff, S. B., et al. 2020, *MNRAS*, **499**, 3178
- Zink, B. 2011, arXiv:1102.5202