The Einstein Toolkit: a community computational infrastructure for relativistic astrophysics

IOP PUBLISHING CLASSICAL AND QUANTUM GRAVITY

Class. Quantum Grav. **29** (2012) 115001 (56pp) doi:10.1088/0264-9381/29/11/115001

The Einstein Toolkit: a community computational infrastructure for relativistic astrophysics

Frank Löffler[1], Joshua Faber[2], Eloisa Bentivegna[3], Tanja Bode[4], Peter Diener[1], Roland Haas[4,5], Ian Hinder[3], Bruno C Mundim[2], Christian D Ott[1,5,6], Erik Schnetter[1,7,8], Gabrielle Allen[1,9,10], Manuela Campanelli[2] and Pablo Laguna[4]

[1] Center for Computation & Technology, Louisiana State University, Baton Rouge, LA, USA
[2] Center for Computational Relativity and Gravitation, School of Mathematical Sciences, Rochester Institute of Technology, Rochester, NY, USA
[3] Max-Planck-Institut für Gravitationsphysik, Albert-Einstein-Institut, Golm, Germany
[4] Center for Relativistic Astrophysics, School of Physics, Georgia Institute of Technology, Atlanta, GA, USA
[5] TAPIR, California Institute of Technology, Pasadena, CA, USA
[6] Institute for the Physics and Mathematics of the Universe, University of Tokyo, Kashiwa, Japan
[7] Perimeter Institute for Theoretical Physics, Waterloo, ON, Canada
[8] Department of Physics, University of Guelph, Guelph, ON, Canada
[9] Department of Computer Science, Louisiana State University, Baton Rouge, LA, USA
[10] National Science Foundation, USA

E-mail: knarf@cct.lsu.edu

Received 28 November 2011, in final form 16 April 2012
Published 8 May 2012
Online at stacks.iop.org/CQG/29/115001

Abstract

We describe the Einstein Toolkit, a community-driven, freely accessible computational infrastructure intended for use in numerical relativity, relativistic astrophysics, and other applications. The toolkit, developed by a collaboration involving researchers from multiple institutions around the world, combines a core set of components needed to simulate astrophysical objects such as black holes, compact objects, and collapsing stars, as well as a full suite of analysis tools. The Einstein Toolkit is currently based on the Cactus framework for high-performance computing and the Carpet adaptive mesh refinement driver. It implements spacetime evolution via the BSSN evolution system and general relativistic hydrodynamics in a finite-volume discretization. The toolkit is under continuous development and contains many new code components that have been publicly released for the first time and are described in this paper. We discuss the motivation behind the release of the toolkit, the philosophy underlying its development, and the goals of the project. A summary of the implemented numerical techniques is included, as are results of numerical test covering a variety of sample astrophysical problems.

PACS numbers: 04.25.D−, 04.30.−w, 04.70.−s, 07.05.Tp, 95.75.Pq

(Some figures may appear in colour only in the online journal)

Contents

1. Introduction 2
2. Requirements 4
   2.1. Scientific 4
   2.2. Academic and social 4
3. Design and strategy 5
4. Core technologies 5
   4.1. Cactus framework 5
   4.2. Adaptive mesh refinement 6
   4.3. Simulation factory 8
   4.4. Kranc 9
5. Components 10
   5.1. Simulation domain, symmetries, boundaries 10
   5.2. Base modules 14
   5.3. Initial data 17
   5.4. Spacetime curvature evolution 22
   5.5. Hydrodynamics evolution 25
   5.6. Equations of state 29
   5.7. Analysis 30
6. Examples 35
   6.1. Spinning BH 35
   6.2. BH binary system 38
   6.3. Linear oscillations of TOV stars 42
   6.4. NS collapse 45
   6.5. Cosmology 46
7. Conclusion and future work 47

Acknowledgments 49

References 49

1. Introduction

Scientific progress in the field of numerical relativity has always been closely tied to the availability and ease-of-use of enabling software and computational infrastructure. This paper describes the Einstein Toolkit, which provides such an infrastructure, developed openly and made available freely with grant support from the National Science Foundation.

Now is a particularly exciting time for numerical relativity and relativistic astrophysics, with major advances having been achieved in the study of astrophysical systems containing black holes (BHs) and neutron stars (NSs). The first fully general relativistic (GR) simulations of merging NS–NS binaries were reported in 1999, with further advances over the next few years [1–5]. However, systems containing BHs proved much more difficult to evolve numerically until 2005. That year, computational breakthroughs were made following the development of a generalized harmonic formulation [6] and then a 'moving puncture' approach [7, 8] in the Baumgarte–Shapiro–Shibata–Nakamura (BSSN) formalism [9, 10] that lead to the first stable long-term evolutions of moving single and multiple BH systems. These results quickly transformed the field which was now able to effectively evolve the Einstein field equations for coalescing BH–BH binaries and other systems containing moving BHs, including merging BH–NS binaries.

These breakthroughs had direct relevance to astrophysics, and enabled exciting new results on recoil velocities from BH–BH mergers (e.g., [11–18] and references therein), post-Newtonian and numerical waveform comparisons and waveform template generation (e.g., [19–29] and references therein), comparisons between numerical waveforms [30–32], determination of the spin of the remnant BH formed in BH–BH mergers (e.g., [33–38] and references therein), and studies of eccentric BH–BH binaries [39–44]. Also, see [45] for a review on BH binaries, gravitational waves and numerical relativity.

Meanwhile, general relativistic magneto-hydrodynamics (GRMHD) on fixed background spacetimes has been successful in multi-dimensional settings since the mid-1990s, focusing on BH accretion processes and relativistic jet production and evolution (see [46] for a review of the numerical formalism and [47] for a review of work on disk and jet models). GRMHD coupled with curvature evolution, on the other hand, which is crucial for modeling large-scale bulk dynamics in compact binary star coalescence or single-star collapse scenarios, has started to produce astrophysically interesting results only in the past ∼3–5 years, enabled primarily by the availability of long-term stable curvature evolution systems as well as improved GRMHD algorithms (see [46] for a review). In addition to these developments, substantial progress has been made in importing more physically motivated equations of state (EOSs), including tabulated versions (e.g., [48–50]) and temperature-dependent models (e.g., [51–53]). Some codes have also begun to incorporate microphysical effects of neutrino emission and deleptonization [54, 55].

Many of the successful techniques used to evolve BH–BH binaries have proven to be equally applicable to merging NS–NS [54, 56–74] and BH–NS [44, 74–90] binaries (for reviews, see also [91, 92]), allowing for further investigations into the former and the first full GR simulations of the latter. All recent results use either the generalized harmonic formalism or the BSSN formalism in the 'moving puncture' gauge. Nearly all include some form of adaptive mesh refinement (AMR), since unigrid models cannot produce accurate long-term evolutions without requiring exorbitant computational resources, though some BH–NS simulations have been performed with a pseudospectral code [76, 77, 80, 81]. Many groups' codes now include GRMHD (used widely for NS–NS mergers, and for BH–NS mergers in [75], and some include microphysical effects as well (e.g., [54, 72, 77]).

In addition to studying binary mergers, numerical relativity is a necessary element for understanding stellar collapse and dynamical instabilities in NSs. General relativistic hydrodynamic (GRHD) has been used to study, among many other applications, massive stars collapsing to protoneutron stars [93–95], the collapse of rotating, hypermassive NSs to BHs in 2D and 3D (see, e.g., [96–102]), and non-axisymmetric instabilities in rapidly rotating polytropic NS models [97, 102, 103].

Simultaneously with the advances in both our physical understanding of relativistic dynamics and the numerical techniques required to study them, a set of general computational tools and libraries has been developed with the aim of providing a computational core that can enable new science, broaden the community, facilitate collaborative and interdisciplinary research, promote software reuse and take advantage of emerging petascale computers and advanced cyberinfrastructure: the `Cactus` computational toolkit [104]. Although the development of `Cactus` was driven directly from the numerical relativity community, it was developed in collaboration with computer scientists and other computational fields to facilitate the incorporation of innovations in computational theory and technology.

This success prompted usage of the `Cactus` computational toolkit in other areas, such as ocean forecast models [105] and chemical reaction simulations [106]. At the same time, the growing number of results in numerical relativity increased the need for commonly available utilities such as comparison and analysis tools, typically those specifically designed for astrophysical problems. Including them within the `Cactus` computational toolkit was not felt to fit within its rapidly expanding scope. This triggered the creation of the Einstein Toolkit [107]. Large parts of the Einstein Toolkit currently do make use of the `Cactus` toolkit, but this is not an requirement, and other contributions are welcome, encouraged and have been accepted in the past.

In the remainder of this paper, we describe the Einstein Toolkit, a collection of freely available and easy-to-use computational codes for numerical relativity and relativistic astrophysics. The code details and example results present in this paper represent the state of the Einstein Toolkit in its release ET_2011_05 'Curie', released on April 21, 2011. In sections 2 and 3 we describe the goals and overall design principles of the Einstein Toolkit. Section 4 provides a high-level overview of the core components used in the Einstein Toolkit. The core components form the basis of the toolkit and define the environment that a toolkit user works in. Section 5, one of the two main section of this paper, contains a detailed description of the major components of the toolkit. Each component's function is explained and, if applicable, the physical system it models is introduced. The next major section, section 6, contains a set of sample results obtained using the toolkit's components. Examples from numerical relativity, astrophysics and cosmology are provided.

2. Requirements

2.1. Scientific

While the aforementioned studies collectively represent breakthrough simulations that have significantly advanced the modeling of relativistic astrophysical systems, all simulations are currently missing one or more critical physical ingredients and are lacking the numerical precision to accurately and realistically model the large-scale and small-scale dynamics of their target systems simultaneously.

One of the aims of the Einstein Toolkit is to provide or extend some of these missing ingredients in the course of its development. Over the past three years, routines have been added to the code to allow for a wider range of initial data choices, to allow for multithreading in hydrodynamic evolutions, and to refine the `Carpet` AMR driver. Looking forward, three possible additions to future releases are the inclusion of magnetic fields into the dynamics via an ideal MHD treatment, more physical nuclear matter EOSs including the ability to model finite-temperature effects, and higher order numerical techniques. All of these are under active development, with MHD and finite-temperature evolution code already available, though not completely documented, within the public toolkit releases, and will be made available once they are thoroughly tested and validated against known results.

2.2. Academic and social

A primary concern for research groups is securing reliable funding to support graduate students and postdoctoral researchers. This is easier to achieve if it can be shown that scientific goals can be attacked directly with fewer potential infrastructure problems, one of the goals of the Einstein Toolkit.

While the Einstein Toolkit does have a large group of users, many of them do not directly collaborate on science problems, and some compete. However, many groups agree that sharing the development of the underlying infrastructure is mutually beneficial for every group and the wider community as well. This is achieved by lifting off the research groups' shoulders much of the otherwise necessary burden of creating such an infrastructure, while at the same time increasing the amount of code review and thus, code quality. In addition, the Einstein Toolkit provides computer scientists an ideal platform to perform state-of-the-art research, which directly benefits research in other areas of science and provides an immediate application of their research.

3. Design and strategy

The mechanisms for the development and support of the Einstein Toolkit are designed to be open, transparent and community-driven. The complete source code, documentation and tools included in the Einstein Toolkit are distributed under open-source licenses. The Einstein Toolkit maintains a version control system (`svn.einsteintoolkit.org`) with open access that contains software supported by the Einstein Toolkit, the toolkit web pages, and documentation. An open wiki for documentation (`docs.einsteintoollkit.org`) has been established where the community can contribute either anonymously or through personal authentication. Almost all discussions about the toolkit take place on an open mail list (`users@einsteintoolkit.org`). The regular weekly meetings for the Einstein Toolkit are open and the community is invited to participate. Meeting minutes are recorded and publicly available as well. The Einstein Toolkit blog requires users to first request a login, but then allows for posting at will. Any user can post comments to entries already on the blog. The community makes heavy use of an issue tracking system (`trac.einsteintoolkit.org`), with submissions also open to the public.

Despite this open design, some actions naturally have to be restricted to a smaller group of maintainers. This is true for administrative tasks like the setup and maintenance of the services themselves, or to avoid large amounts of spam. One of the most important tasks of an Einstein Toolkit maintainer is to review and apply patches sent by users in order to ensure a high software quality level. Every substantial change or addition to the toolkit must be reviewed by another Einstein Toolkit maintainer, and is generally open for discussion on the users mailing list. This convention, though not being strictly enforced, works well in practice and promotes active development.

4. Core technologies

The Einstein Toolkit modules center around a set of core modules that provide basic functionality to create, deploy and manage a numerical simulation starting with code generation all to way to archiving of simulation results: (i) the `Cactus` framework 'flesh' provides the underlying infrastructure to build complex simulation codes out of independently developed modules and facilitates communication between these modules. (ii) The AMR driver, `Carpet`, is built on top of `Cactus` and provides problem independent AMR support for simulations that need to resolve physics on length scales differing by many orders of magnitude, while relieving the scientist of the need to worry about internal details of the mesh refinement driver. (iii) `Kranc`, which generates code in a computer language from a high-level description in Mathematica and (iv) the simulation factory, which provides a uniform, high-level interface to common operations, such as submission and restart of jobs, for a large number of compute clusters.

4.1. Cactus framework

The `Cactus` framework [104, 108, 109] is an open-source, modular, portable programming environment for collaborative high-performance computing (HPC) primarily developed at

Louisiana State University. Cactus originated at the Albert Einstein Institute and also has roots at the National Center for Supercomputing Applications (see, e.g., [110–112] for historical reviews). The Cactus computational toolkit consists of general modules which provide parallel drivers, coordinates, boundary conditions, interpolators, reduction operators, and efficient I/O in different data formats. Generic interfaces make it possible to use external packages and improved modules which are made immediately available to users.

The structure of the Cactus framework is completely modular, with only a very small core (the 'flesh') which provides the interfaces between modules both at compile- and run-time. The Cactus modules (called 'thorns') may (and typically do) specify inter-module dependences, e.g., to share or extend configuration information, common variables, or runtime parameters. Modules compiled into an executable can remain dormant at run-time. This usage of modules and a common interface between them enables researchers to (1) easily use modules written by others without the need to understand all details of their implementation and (2) write their own modules without the need to change the source code of other parts of a simulation in the (supported) programming language of their choice. The number of active modules within a typical Cactus simulation ranges from tens to hundreds and often has an extensive set of inter-module dependences.

The Cactus framework was developed originally by the numerical relativity community, and although it is now a general component framework that supports different application domains, its core user group continues to be comprised of numerical relativists. It is not surprising therefore, that one of the science modules provided in the Einstein Toolkit is a set of state-of-the-art modules to simulate binary BH mergers. All modules to simulate and analyze the data are provided out of the box. This set of modules also provides a way of testing the Einstein Toolkit modules in a production type simulation rather than synthetic test cases. Some of these modules have been developed specifically for the Einstein Toolkit while others are modules used in previous publications and have been contributed to the toolkit. In these cases the Einstein Toolkit provides documentation and best practice guidelines for the contributed modules.

4.2. Adaptive mesh refinement

In Cactus, infrastructure capabilities such as memory management, parallelization, time evolution, mesh refinement, and I/O are delegated to a set of special *driver* components. This helps separate physics code from infrastructure code; in fact, a typical physics component (implementing, e.g., the Einstein or relativistic MHD equations) does not contain any code or subroutine calls having to do with parallelization, time evolution, or mesh refinement. The information provided in the interface declarations of the individual components allows a highly efficient execution of the combined program. Cactus's parallelization paradigm is based on a spatial domain decomposition, and is briefly explained in figure 1.

The Einstein Toolkit offers two drivers, PUGH and Carpet. PUGH provides domains consisting of a uniform grid with Cartesian topology, and is highly scalable (up to more than 130 000 cores [113]). Carpet [114–116] provides multi-block methods and AMR. Multi-block methods cover the domain with a set of (possibly distorted) blocks that exchange boundary information via techniques such as interpolation or penalty methods[11]. The AMR capabilities employ the standard Berger–Oliger algorithm [117] with subcycling in time.

AMR implies that resolution in the simulation domain is dynamically adapted to the current state of the simulation, i.e. regions that require a higher resolution are covered with

[11] Although multi-block methods are supported by Carpet, the Einstein Toolkit itself does not currently contain any multi-block coordinate systems.

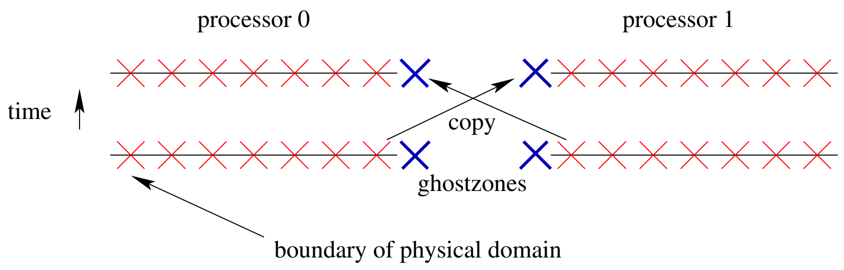


Figure 1. Cactus employs spatial domain decomposition to distribute workload and storage across processors. It stores *ghost zones* (additional, 'dummy' grid points, here shown as bold and blue crosses) at inter-process boundaries to allow evaluating computational stencils near these boundaries. After modifying data, these ghost zones need to be *synchronized*, which requires inter-processor communication. This is handled by a special *driver* component (see main text).

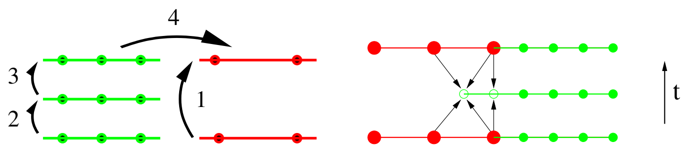


Figure 2. Berger–Oliger time stepping details, showing a coarse and a fine grid, with time advancing upward. *Left:* time stepping algorithm. First the coarse grid takes a large time step, then the refined grid takes two smaller steps. The fine grid solution is then injected into the coarse grid where the grids overlap. *Right:* fine grid boundary conditions. The boundary points of the refined grids are filled via interpolation. This may require interpolation in space and in time.

blocks with a finer grid (typically by a factor of 2); these are called *refinement levels*. Finer grids can be also recursively refined. At regular intervals, the resolution requirements in the simulation are re-evaluated, and the grid hierarchy is updated; this step is called *regridding*.

In most simulations using Carpet the sizes of refinement levels are predetermined by the user, while their locations are calculated adaptively to track the locations of interesting features such as BHs or stars. In addition, refinement levels may be disabled or activated depending on the dynamics of a particular simulation, responding, e.g., to events such as the formation of a common horizon or an increasing stellar density due to collapse, respectively.

Since a finer grid spacing also requires smaller time steps for hyperbolic problems, the finer grids perform multiple time steps for each coarse grid time step, leading to a recursive time evolution pattern that is typical for Berger–Oliger AMR. If a simulation uses 11 levels, then the resolutions (both in space and time) of the the coarsest and finest levels differ by a factor of $2^{11-1} = 1024$. This non-uniform time stepping leads to a certain complexity that is also handled by the `Carpet` driver; for example, applying boundary conditions to a fine level requires interpolation in space and time from a coarser level (interpolation from a coarse level to a finer level is referred to as 'prolongation' within discussions of AMR; the converse operation is 'restriction') . Outputting the solution at a time in between coarse grid time steps also requires interpolation. These parallel interpolation operations are implemented efficiently in `Carpet` and are applied automatically as specified in the execution schedule, i.e. without requiring function calls in user code. Figure 2 describes some details of the Berger–Oliger time stepping algorithm; more details are described in [114].

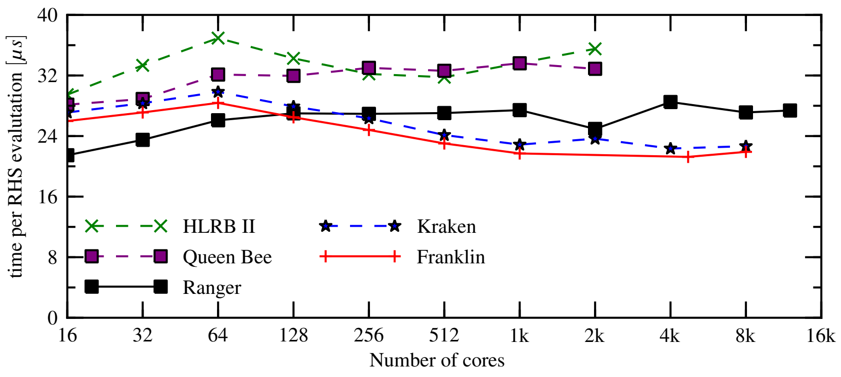


Figure 3. Results from weak scaling tests evolving the Einstein equations on a mesh refinement grid structure with nine levels. This shows the time required per grid point, where smaller numbers are better (the ideal scaling is a horizontal line). This demonstrates excellent scalability to up to more than 10 000 cores. Including hydrodynamics approximately doubles calculation times without negatively influencing scalability. The characteristics of the systems used in this benchmark are described in table 1.

Table 1. Characteristics of the systems used in the benchmark figure 3. These systems represent a wide range of HPC architectures that were available over the past years.

Name	Architecture (CPU)	Interconnect	Nodes	Cores/node	CPU freq. (GHz)
Franklin (NERSC)	Cray XT4 (AMD)	SeaStar2	8502	4	2.3
HLRB II (LRZ Munich)	SGI Altix (Itanium)	NUMAlink	1	9728	1.6
Kraken (NICS)	Cray XT5 (AMD)	SeaStar	9408	12	2.6
Queen Bee (LONI)	Intel cluster	InfiniBand	668	8	2.33
Ranger (TACC)	AMD cluster	InfiniBand	1888	16	2.3

`Carpet` is the main driver used today for `Cactus`-based astrophysical simulations. `Carpet` offers hybrid MPI/OpenMP parallelization and is used in production on up to several thousand cores [118, 119]. Figure 3 shows a weak scaling test of `Carpet`, where `McLachlan` (see section 5.4 below) solves the Einstein equations evolving a Minkowski spacetime on a grid structure with nine levels of mesh refinement. This demonstrates excellent scalability up to more than 10 000 cores. In production simulations, smaller and more complex grid structures, serial tasks related to online data analysis and other necessary tasks reduce scalability by up to a factor of 10.

We estimate that in 2010, about 7000 core years of computing time (45 million core h) were used via `Carpet` by more than a dozen research groups world-wide. To date, more than 90 peer-reviewed publications and more than 15 student theses have been based on `Carpet` [116].

4.3. Simulation factory

Today's supercomputers differ significantly in their hardware configuration, available software, directory structure, queuing system, queuing policy, and many other user-visible properties. In addition, the system architectures and user interfaces offered by supercomputers are very different from those offered by laptops or workstations. This makes performing large, three-dimensional, time-dependent simulations a complex, time-consuming and difficult task. However, most of these differences are only superficial, and the basic capabilities of supercomputers are very similar; most of the complexity of managing simulations lies in menial tasks that require no physical or numerical insight.

The simulation factory [120, 121] offers a set of abstractions for the tasks necessary to set up and successfully complete numerical simulations based on the `Cactus` framework. These abstractions hide tedious low-level management operations, capture 'best practices' of experienced users, and create a log trail ensuring repeatable and well-documented scientific results. Using these abstractions, most operations are simplified and many types of potentially disastrous user errors are avoided, allowing different supercomputers to be used in a uniform manner.

Using the simulation factory, we offer a tutorial for the Einstein Toolkit [107] that teaches new users how to download, configure, build, and run full simulations of the coupled Einstein/relativistic hydrodynamics equations on a supercomputer with a few simple commands. Users need no prior knowledge about either the details of the architecture of a supercomputer nor its particular software configuration. The same exact set of SimFactory commands can be used on all other supported supercomputers to run the same simulation there.

The simulation factory supports and simplifies three kinds of operations:

1. *Remote access.* The actual access commands and authentication methods differ between systems, as do the user names that a person has on different systems. Some systems are not directly accessible, and one must log in to a particular 'trampoline' server first. The simulation factory hides this complexity.
2. *Configuring and building.* Building `Cactus` requires certain software on the system, such as compilers, libraries, and build tools. Many systems offer different versions of these, which may be installed in non-default locations. Finding a working combination that results in efficient code is extremely tedious and requires low-level system experience. The simulation factory provides a *machine database* that enables users to store and exchange this information. In many cases, this allows people to begin to use a new machine in a very short time with just a few, simple commands.
3. *Submitting and managing simulations.* Many simulations run for days or weeks, requiring frequent checkpointing and job re-submission because of short queue run-time limits. Simple user errors in these menial tasks can potentially destroy weeks of information. The simulation factory offers safe commands that encapsulate best practices that prevent many common errors and leave a log trail.

The above features make running simulations on supercomputers much safer and simpler.

4.4. Kranc

`Kranc` [122–124] is a Mathematica application which converts a high-level continuum description of a PDE into a highly optimized module for `Cactus`, suitable for running on anything from a laptop to the world's largest HPC systems. Many codes contain a large amount of complexity, including expanded tensorial expressions, numerical methods, and the large amount of 'glue' code needed for interfacing a modern HPC application with the underlying framework. `Kranc` absorbs this complexity, allowing the scientist to concentrate on writing only the `Kranc` script which describes the continuum equations.

This approach brings with it many advantages. With these complicated elements factored out, a scientist can write many different `Kranc` codes, all taking advantage of the features of

Kranc and avoiding unnecessary or trivial but painstaking duplication. The codes might be variations on a theme, perhaps versions which use different sets of variables or formulations of the equations, or they could represent completely different physical systems. The use of a symbolic algebra package, Mathematica, enables high-level optimizations which are not performed by the compiler to be implemented in Kranc.

Any enhancements to Kranc can be automatically applied to all codes which are generated using Kranc. Existing codes have easily benefited from the following features added to Kranc after the codes themselves were written: (i) OpenMP parallelization support, necessary for efficient use of modern multi-core processors; (ii) support for multipatch domains with the Llama [125] code; (iii) automatic generation of vectorized code, where the equations are evaluated simultaneously by the processor for two grid points at the same time; and (iv) common sub-expression elimination, and various other optimization strategies.

Within the Einstein Toolkit, the Einstein evolution thorn McLachlan, as well as the wave extraction thorn WeylScal4, are both generated using Kranc, and hence support all the above features.

5. Components

The Einstein Toolkit uses the modular Cactus framework as its underlying infrastructure. A simulation within Cactus could just use one module, but in practice simulations are often composed from hundreds of components. Some of these modules construct grid domains and their properties (see section 5.1) or provide common definitions and conventions (see section 5.2). Others provide initial data (see section 5.3), which may be evolved using the different evolution methods for vacuum and matter configurations described in sections 5.4 and 5.5, respectively. The thermodynamic properties of fluids are encoded in EOSs (see section 5.6). Finally, additional quantities which are not directly evolved are often interesting for a detailed analysis of the simulation's results. Modules providing commonly used analysis methods are described in section 5.7. To aid the reader, we summarize in table 2 the key mathematical symbols defined throughout this paper along with the module in which the corresponding quantity is stored in the code.

5.1. Simulation domain, symmetries, boundaries

5.1.1. Domains and coordinates. Cactus distinguishes between the *physical* domain, which lives in the continuum, and *discrete* domain, which consists of a discrete set of grid points. The physical domain is defined by its coordinate extent and is independent of the numerical resolution; in particular, the boundary of the physical domain has a width of zero (and is thus a set of measure zero). The discrete domain is defined indirectly via a discretization procedure that specifies the number of boundary points, their location with respect to the physical boundary, and either the grid spacing or the number of grid points spanning the domain. This determines the number and location of the grid points in the discrete domain. The discrete domain may have grid points outside of the physical domain, and may have a non-zero boundary width. This mechanism ensures that changes in the numerical resolution do not affect the extent of the physical domain, i.e. that the discrete domains converge to the physical domain in the limit of infinite resolution.

The Einstein Toolkit provides the CoordBase thorn that facilitates the definition of the simulation domain independent of the actual evolution thorn used, allowing it to be specified at run time via parameters in the same way that parameters describing the physical system are specified. CoordBase exposes a public runtime interface that allows other thorns to query

Table 2. Summary of key variables used throughout this paper, along with the equations that define them and the Einstein Toolkit thorn and group name used to store the quantity, where applicable.

Symbol	Quantity	Equation	Einstein Toolkit thorn::group
$G_{\mu\nu}$	Einstein tensor	(1)	N/A
$T_{\mu\nu}$	Stress–energy tensor	(6)	TmunuBase::stress_energy_tensor
$g_{\mu\nu}$	Spacetime 4-metric	(2)	N/A
$F_{\mu\nu}$	Faraday tensor	(5)	N/A
u^μ	4-velocity	N/A	N/A
γ_{ij}	Spatial 3-metric	(2)	ADMBase::metric
α	Lapse function	(2)	ADMBase::lapse
β^i	Shift vector	(2)	ADMBase::shift
n^μ,n_μ	Unit normal	N/A	N/A
K_{ij}	Extrinsic curvature	(3)	ADMBase::curv
ρ	Rest mass density	(6), (7)	HydroBase::rho
P	Fluid pressure	(7)	HydroBase::press
ϵ	Specific internal energy	(7)	Hydrobase::eps
h	Specific enthalpy	(7)	N/A
v^i	3-velocity	(4)	HydroBase::vel
B^i	Magnetic field vector	(5)	HydroBase::Bvec
ψ, (ϕ)	(Logarithmic) conformal factor	(10), (21)	(ML_BSSN::ML_log_confac)
$\tilde{\gamma}_{ij}$	Conformal 3-metric	(22)	ML_BSSN::ML_metric
K	Trace of extrinsic curvature	(23)	ML_BSSN::ML_trace_curv
$\tilde{A}_{ij}$	Conformal extrinsic curvature	(24)	ML_BSSN::ML_curv
$\tilde{\Gamma}^i$	Conformal connection	(25)	ML_BSSN::ML_Gamma
D	Conservative density	(54)	GRHydro::dens
S_i	Conservative momentum density	(55)	GRHydro::scon
τ	Conservative energy density	(56)	GRHydro::tau
W	Lorentz factor	(51)	GRHydro::w_lorentz

the domain description in a uniform way. This is used by `Carpet` to query `CoordBase` for the discrete grid when creating the hierarchy of grids, automatically ensuring a consistent grid description between the two thorns. Evolution thorns such as `McLachlan` use the domain information to decide which points are evolved and therefore require the evaluation of the right-hand-side (RHS) expression, and which ones are set via boundary or symmetry conditions.

5.1.2. Symmetries and boundary conditions. The Einstein Toolkit includes two thorns, `Boundary` and `SymBase`, to provide a generic interface to specify and implement boundary and symmetry conditions. The toolkit includes built-in support for a set of reflecting or rotating symmetry conditions that can be used to reduce the size of the simulation domain. These symmetries include periodicity in any of the coordinate directions (via the `Periodic` module), reflections across the coordinate planes (via the `Reflection` module), 90° and 180° rotational symmetries (via the `RotatingSymmetry90` and `RotatingSymmetry180` modules, respectively) about the z-axis, and a continuous rotational symmetry (via the `Cartoon2D` thorn) [126]. `Cartoon2D` allows fully three-dimensional codes to be used in axisymmetric problems by evolving a slice in the $y = 0$ plane and using the rotational symmetry to populate boundary points off the plane (see figure 4).

In applying symmetries to populate boundary zones, the transformation properties of tensorial quantities (including tensor densities and non-tensors such as Christoffel symbols) are correctly taken into account, just as they are in the interpolation routines present in `Cactus`. Thus, symmetries are handled transparently from the point of view of user modules (see figure 5 for an illustration).

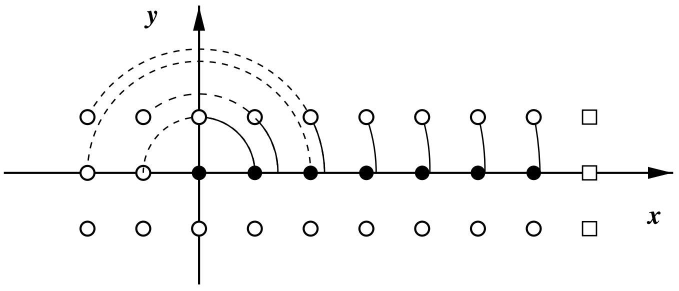


Figure 4. Grid layout of a simulation using `Cartoon2D`. The z-axis is the axis of rotational symmetry. Only points on the non-negative x-axis (full circles) are evolved, all other points (hollow circles) are filled via interpolation along the x-axis. The dashed circle sections describe how points are rotated onto the x-axis. The hollow squares on the right denote boundary points that cannot by set by this algorithm. (Image reproduced from [126], figure 1.)

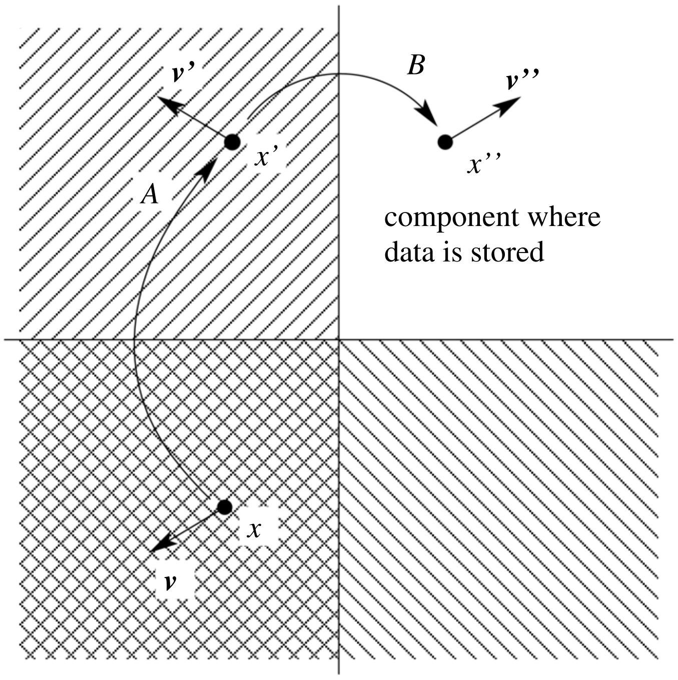


Figure 5. Iterative transformation of a point x in quadrant 3 to the corresponding point x'' for which there is actual data stored. In this example, two reflection symmetries along the horizontal and vertical axis are present. Notice how the vector components change in transformations A and B.

The `Boundary` thorn serves as a registry for available boundary conditions and provides basic scheduling to enforce all requested boundary conditions at the proper times. It also provides a basic set of boundary conditions to be used by user thorns. The 'flat' boundary conditions often used for hydrodynamic variables that approach an atmosphere value fall in this category. More complicated boundary conditions are often implemented as modifications to the evolution equations and are not handled directly by `Boundary`. Examples are the radiative (Sommerfeld) and extrapolation boundary conditions provided by thorn `NewRad`.

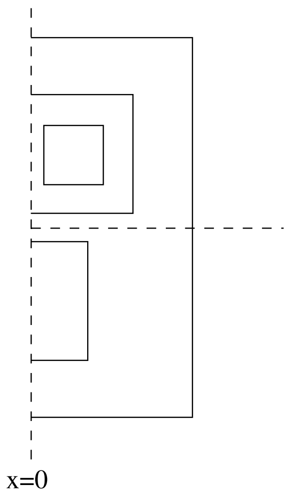


Figure 6. Nested boxes following the individual BHs in an equal-mass binary BH merger simulation (see section 6.2), with the location of the individual BHs found by PunctureTracker. The innermost three of the nine levels of mesh refinement used in this simulation are shown. Notice the use of RotatingSymmetry180 to reduce the computational domain.

5.1.3. Adaptive mesh refinement. The Einstein Toolkit currently supports feature-based mesh refinement, which is based on extracting quantities such as the locations of BHs or NSs and then constructing a mesh hierarchy (stacks of refined regions) based on the locations, sizes, and speeds of these objects. This allows tracking objects as they move through the domain. One can also add or remove stacks if, for instance, the number of objects changes. Full AMR based on a local error estimate is supported by Carpet, but the Einstein Toolkit does not currently provide a suitable regridding thorn to create such a grid. If initial conditions are constructed outside of Carpet (which is often the case), then the initial mesh hierarchy has to be defined manually. In order to facilitate the description of the mesh hierarchy the Einstein Toolkit provides two regridding modules in the CarpetRegrid and CarpetRegrid2 thorns. Both thorns primarily support box-in-box type refined meshes, which are well suited to current binary BH simulations in which the high-resolution regions are centered on the individual BHs. Figure 6 shows a typical set of nested boxes during the inspiral phase of an equal-mass binary BH merger simulation.

If one wants to exploit a particular non-local symmetry of the domain (e.g. a rotational symmetry, or periodicity in some directions), then the grid structure also needs to have this symmetry. CarpetRegrid2 thus contains code to enforce such symmetries on the grid structures it creates (see figure 7).

CarpetRegrid provides a number of different ways to specify the refined regions, e.g., as a set of boxes centered around the origin or as an explicit list of regions that make up the grid hierarchy. Traditionally, groups using CarpetRegrid have employed auxiliary thorns that are not part of the Einstein Toolkit to create this list of boxes based on information obtained from apparent horizon (AH) tracking or other means. CarpetRegrid2 provides a user-friendly interface to define sets of nested boxes that follow the location of BHs or other tracked objects. Object coordinates are updated by CarpetTracker, which provides a simple interface to the object trackers PunctureTracker and NSTracker (see section 5.7.4) in order to have the refined region follow the moving objects.

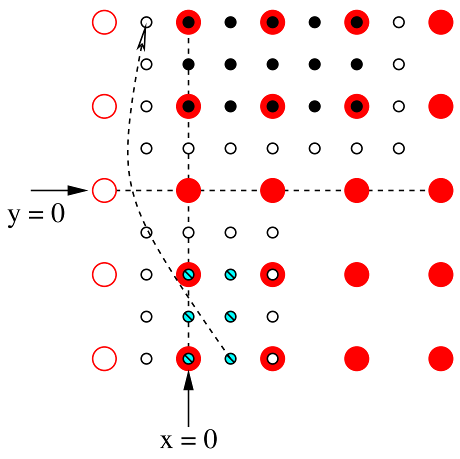


Figure 7. Example of a grid layout created by `CarpetRegrid2`. This figure shows two refinement levels, a coarse (big red circles) and a fine one (small black circles). In this example we use one boundary point and one ghost point, as well as `RotatingSymmetry180`. The boundary points are filled by the symmetry condition, the ghost points are filled via interpolation from the coarse grid. Starting from a user-specified refined region consisting of 5×3 points (small, dark, filled circles in the upper half), `CarpetRegrid2` enforced the the π-symmetry by adding the 2×3 block of refined points in the lower half (small, light, slashed circles). The open circles are boundary and ghost points maintained by `Carpet`. The arrow demonstrates how the π-symmetry fills in a boundary point.

5.2. Base modules

Modular designs have proven to be essential for distributed development of complex software systems and require the use of well-defined interfaces. Low-level interoperability within the Einstein Toolkit is provided by the `Cactus` infrastructure. One example of this is the usage of one module from within another, e.g., by calling a function within another thorn independent of programming language used for both the calling and called function. Solutions for technical challenges like this can be and are provided by the underlying framework, in this case `Cactus`.

However, certain other standards are very hard or impossible to enforce on a technical level. Examples for these include the exact definitions of physical variables, their units, and, on a more technical level, the variable names used for the physical quantities. Even distinct simulation codes typically use very similar scheduling schemes, so conventions describing the behavior of the scheduler can help coordinate the order in which functions in different modules are called.

The Einstein Toolkit provides modules whose sole purpose is to declare commonly used variables and define their meaning and units. These conditions are not strictly enforced, but instead documented for the convenience of the user community. Three of these base modules, `ADMBase`, `HydroBase`, and `TmunuBase`, are described in more detail below.

In the following, we assume that the reader is familiar with the basics of numerical relativity and GR hydrodynamics, including the underlying differential geometry and tensor analysis. Detailed introductions to numerical relativity have recently been given by Alcubierre [127], Baumgarte and Shapiro [128], and Centrella *et al* [45]. GR hydrodynamics has been reviewed by Font [46]. We set $G = c = 1$ throughout this paper, and $M_\odot = 1$ where appropriate.

5.2.1. ADMBase. The Einstein Toolkit provides code to evolve the Einstein equations

$$G^{\mu\nu} = 8\pi T^{\mu\nu}, \tag{1}$$

where $G^{\mu\nu}$ is the Einstein tensor, describing the curvature of four-dimensional spacetime, and $T^{\mu\nu}$ is the stress–energy tensor. Relativistic spacetime evolution methods used within the `Cactus` framework employ different formalisms to accomplish this goal, but essentially all are based on the 3+1 Arnowitt–Deser–Misner (ADM) construction [129, 130], which makes it the natural choice of a common foundation for exchanging data between modules using different formalisms. In the 3 + 1 approach, four-dimensional spacetime is foliated into sequences of spacelike three-dimensional hypersurfaces (slices) connected by timelike normal vectors. The 3 + 1 split introduces four gauge degrees of freedom: the lapse function α that describes the advance of proper time with respect to coordinate time for a normal observer[12] and the shift vector β^i that describes how spatial coordinates change from one slice to the next.

Within the ADM formulation the spacetime metric is written in the form

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

where $g_{\mu\nu}$, α, β^i, and γ_{ij} are the spacetime 4-metric, lapse function, shift vector, and spatial 3-metric, respectively, and we follow the standard relativistic convention where Latin letters are used to index three-dimensional spatial quantities and Greek letters to index four-dimensional spacetime quantities, with the index running from 0 to 3. The remaining dynamical component of the spacetime is contained in the definition of the extrinsic curvature K_{ij}, which is defined in terms of the time derivative of the metric after incorporating a Lie derivative with respect to the shift vector:

$$K_{ij} \equiv -\frac{1}{2\alpha}(\partial_t - \mathcal{L}_\beta)\gamma_{ij}. \tag{3}$$

The 3-metric, extrinsic curvature, lapse function, and shift vector are all declared as variables in the `ADMBase` module, the latter two together with their first time derivatives. The variables provided by `ADMBase` are:

- The 3-metric tensor, γ_{ij}: `gxx`, `gxy`, `gxz`,`gyy`, `gyz`, `gzz`
- The extrinsic curvature tensor, K_{ij}: `kxx`, `kxy`, `kxz`, `kyy`, `kyz`, `kzz`
- The lapse function, α: `alp`
- The shift vector β^i: `betax`, `betay`, `betaz`

This base module also defines common parameters to manage interaction between different modules. Examples are the type of requested initial data or the used evolution method.

The type of initial data chosen for a simulation is specified by the parameters `initial_data` (3-metric and extrinsic curvature), `initial_lapse`, `initial_shift`. The time derivatives of the gauge variables (the lapse and shift) are set by the parameters `initial_dtlapse` and `initial_dtshift`, respectively. By default, `ADMBase` initializes the components of the 4-metric (3-metric, extrinsic curvature, lapse, and shift) as instantaneously Minkowski in a standard Cartesian coordinate system: $\gamma_{ij} = \delta_{ij}$ (the Kronecker delta), $K_{ij} = 0$, $\alpha = 1$, $\beta^i = 0$. Initial data thorns override these defaults by extending the parameters.

Analogous to specifying initial data, evolution methods are chosen by the parameters `evolution_method` (3-metric and extrinsic curvature), `lapse_evolution_method`, `shift_evolution_method`, `dtlapse_evolution_method` and `dtshift_evolution_method`. `ADMBase` does not evolve the 3-metric or extrinsic

[12] A normal observer follows a worldline tangent to the unit normal on the 3-hypersurface.

curvature, and holds the lapse and shift static. Evolution thorns extend the ranges of these parameters and contain the evolution code.

The variables defined in ADMBase typically are not used for the actual evolution of the curvature. Instead, it is expected that every evolution module converts its internal representation to the form defined in ADMBase after each evolution step. This procedure enables modules which perform analysis on the spacetime variables to use the ADMBase variables without direct dependence on any of the existing curvature evolution methods.

5.2.2. HydroBase. Similar to `ADMBase`, the module `HydroBase` defines a common basis for interactions between modules of a given evolution problem, in this case relativistic hydrodynamics. `HydroBase` extends the Einstein Toolkit to include an interface within which magnetohydrodynamics may work. `HydroBase`'s main function is to store variables which are common to most if not all hydrodynamics codes solving the Euler equations, the so-called primitive variables. These are also the variables which are needed to couple to a spacetime solver, and often by analysis thorns as well. As with ADMBase, the usage of a common set of variables by different hydrodynamics codes creates the possibility of sharing parts of the code, e.g., initial data solvers or analysis routines. `HydroBase` also defines commonly needed parameters and schedule groups for the main functions of a hydrodynamics code. `HydroBase` uses a set of conventions known as the Valencia formulation [131–133]. In particular, `HydroBase` defines the primitive variables (see [46] for details):

- `rho`: rest mass density ρ
- `press`: pressure P
- `eps`: specific internal energy ϵ
- `vel[3]`: contravariant fluid 3-velocity v^i defined as
$$v^i = \frac{u^i}{\alpha u^0} + \frac{\beta^i}{\alpha} \tag{4}$$
in terms of the 4-velocity u^μ, lapse, and shift vector.
- `Y_e`: electron fraction Y_e
- `temperature`: temperature T
- `entropy`: specific entropy per particle s
- `Bvec[3]`: contravariant magnetic field vector defined as
$$B^i = \frac{1}{\sqrt{4\pi}} n_\nu F^{*\nu i} \tag{5}$$
in terms of the dual $F^{*\mu\nu} = \frac{1}{2}\varepsilon^{\mu\nu\alpha\beta}F_{\alpha\beta}$ to the Faraday tensor and the unit normal of the foliation of spacetime $n^\mu \equiv \alpha^{-1}[1, -\beta^i]^{\mathrm{T}}$.

`HydroBase` also sets up scheduling blocks that organize the main functions which modules of a hydrodynamics code may need. All of those scheduling blocks are optional, but when used they simplify existing codes and make them more interoperable. `HydroBase` itself does not schedule routines inside most of the groups that it provides. Currently the scheduling blocks are:

- Initializing the primitive variables.
- Converting primitive variables to conservative variables.
- Calculating the RHS in the method of lines (MoL).
- Setting and updating an excision mask.
- Applying boundary conditions.

Through these, the initialization of the primitive variables, methods to recover the conservative variables, and basic atmosphere handling can be implemented in different thorns while allowing a central access point for analysis thorns.

5.2.3. TmunuBase. In the Einstein Toolkit, we typically choose the stress–energy tensor $T^{\mu\nu}$ to be that of an ideal relativistic fluid,

$$T^{\mu\nu} = \rho h u^{\mu} u^{\nu} + g^{\mu\nu} P \ , \tag{6}$$

where ρ, u^{μ}, and $g^{\mu\nu}$ are defined above, and

$$h = 1 + \epsilon + P/\rho \tag{7}$$

is the relativistic specific enthalpy.

The thorn `TmunuBase` provides grid functions for the stress–energy tensor $T_{\mu\nu}$ as well as schedule groups to manage when $T_{\mu\nu}$ is calculated. In a simulation, many different thorns may contribute to the stress–energy tensor and this thorn allows them to do so without explicit interdependence. The resulting stress–energy tensor can then be used by the spacetime evolution thorn (again without explicit dependence on any matter thorns). When thorn `MoL` is used for time evolution this provides a high-order spacetime–matter coupling.

The grid functions provided by `TmunuBase` are:

- The time component T_{00}: `eTtt`.
- The mixed components T_{0i}: `eTtx`, `eTty`, `eTtz`.
- The spatial components T_{ij}: `eTxx`, `eTxy`, `eTxz`, `eTyy`, `eTyz`, `eTzz`.

In addition, the grid scalar `stress_energy_state` has the value 1 if storage is provided for the stress–energy tensor and 0 if not.

Thorn `ADMCoupling` provides a similar (but older) interface between spacetime and matter thorns. However, since it is based on an include file mechanism it is more complicated to use. We recommend all new matter thorns to use `TmunuBase` instead.

5.3. Initial data

The Einstein Toolkit contains many modules used to generate initial data for GR simulations, including both vacuum and hydrodynamic configurations. These include modules used primarily for testing of various components, as well as physically motivated configurations that describe, for example, single or binary BHs and/or NSs. Many of the modules are self-contained, consisting of either all the code to generate exact initial solutions or the numerical tools required to construct solutions known semi-analytically. Others, though, require the installation of numerical software packages that are included in the toolkit as external libraries. One example is the `TwoPunctures` module [134]—commonly used in numerical relativity to generate BH–BH binary initial data—which makes use of the GNU scientific library [135, 136]. Several modules have also been implemented to read in data files generated by the `Lorene` code [137, 138].

Initial data setup is in most cases clearly separated from the evolution that follows. Typically, initial data routines provide the data in terms of the quantities defined in the base modules (see section 5.2), and the evolution modules will later convert these quantities to forms used for the evolution. For example, an initial data module must supply g_{ij}, the spatial 3-metric, and K_{ij}, the extrinsic curvature. The conversion between the physical metric and extrinsic curvature and conformal versions of these is handled solely within evolution modules, which are responsible for calculating the conformally related 3-metric $\tilde{\gamma}_{ij}$, the conformal factor ψ or its logarithm ϕ, the conformal traceless extrinsic curvature $\tilde{A}_{ij}$ and the trace of the extrinsic curvature K, as well as initializing the BSSN variable $\tilde{\Gamma}^{i}$ should that be the evolution formalism chosen (see section 5.4 for definitions of these). Optionally, many initial data modules also supply values for the lapse and shift vector and in some cases their time derivatives. It is important to note, though, that many dynamical calculations run better from initial gauge choices set by ansatz rather than those derived from stationary approximations that are incompatible with the gauge evolution equations. In particular, conformal thin-sandwich initial data for binaries include solutions for the lapse and shift that are frequently replaced by simpler analytical values that lead to more circular orbits under standard 'moving puncture' gauge conditions (see, e.g., [78, 139] and other works).

We turn our attention next to a brief discussion of the capabilities of the aforementioned modules as well as their implementation.

5.3.1. Simple vacuum initial data. Vacuum spacetime tests in which the constraint equations are explicitly violated are provided by `IDConstraintViolate` and `Exact`, a set of exact spacetimes in various coordinates including Lorentz-boosted versions of traditional solutions. Vacuum gravitational wave configurations can be obtained by using either `IDBrillData`, providing a Brill wave spacetime [140]; or `IDLinearWaves`, for a spacetime containing a linear gravitational wave. Single BH configurations include `IDAnalyticBH` which generates various analytically known BH configurations; as well as `IDAxibrillBH`, `IDAxiOddBrillBH`, `DistortedBHIVP` and `RotatingDBHIVP`, which introduce perturbations to isolated BHs.

5.3.2. Hydrodynamics tests. Initial data to test different parts of hydrodynamics evolution systems are provided by `GRHydro_InitData`. This module includes several shock tube problems that may be evolved in any of the Cartesian directions or diagonally. All of these have been widely used throughout the field to evaluate a diverse set of solvers [141]. Conservative-to-primitive variable conversion and vice versa are also supported, as are tests to check on the reconstruction of hydrodynamical variables at cell faces (see section 5.5 for more on this). Along similar lines, the `Hydro_InitExcision` module sets up masks for different kinds of excised regions, which is convenient for hydrodynamics tests.

5.3.3. TwoPunctures: binary BHs and extensions. A substantial fraction of the published work on the components of the Einstein toolkit involves the evolution of BH–BH binary systems. The most widely used routine to generate initial data for these is the `TwoPunctures` code (described originally in [134]) which solves the binary puncture equations for a pair of BHs [142]. `TwoPunctures` is a serial code that does not parallelize beyond a single node through the use of `OpenMP`. Nevertheless it has proven sufficiently robust and fast that the vacuum initial data problem is usually solved on-line at the beginning of an inspiral simulation. This approach allows all information required to set up the run to be kept in the single `Cactus` parameter file. `TwoPunctures` assumes the conformal extrinsic curvature $\tilde{K}_{ij}$ to correspond to the Bowen–York form for each BH [143]:

$$\tilde{K}^{ij}_{(m)} = \frac{3}{2r^2}\left(p^i_{(m)}\hat{N}^j + p^j_{(m)}\hat{N}^i - (\tilde{\gamma}^{ij} - \hat{N}^i\hat{N}^j)p^k_{(m)}\hat{N}_k\right) + \frac{3}{r^3}\left(\varepsilon^{ikl}S^{(m)}_k\hat{N}_l\hat{N}^j + \varepsilon^{jkl}S^{(m)}_k\hat{N}_l\hat{N}^i\right), \tag{8}$$

where the sub/superscript (m) refers to the contribution from BH $m = 1, 2$; the 3-momentum is p^i; the BH spin angular momentum is S_i; and $\hat{N}^i = x^i/r$ is the Cartesian normal vector relative to the position of each BH in turn. The conformal extrinsic curvature is raised and lowered using the conformal metric $\tilde{\gamma}_{ij}$ (see (22)) and is related to the physical extrinsic curvature by the conditions

$$\tilde{K}_{ij} = \psi^2 K_{ij}; \qquad \tilde{K}^{ij} = \psi^{10} K_{ij}, \tag{9}$$

where the conformal ψ is defined as

$$\psi \equiv |\det \gamma_{ij}|^{1/12} = \mathrm{e}^{\phi} \qquad (\text{so that } \gamma_{ij} = \psi^4\tilde{\gamma}_{ij}, \det\tilde{\gamma}_{ij} = 1) \tag{10}$$

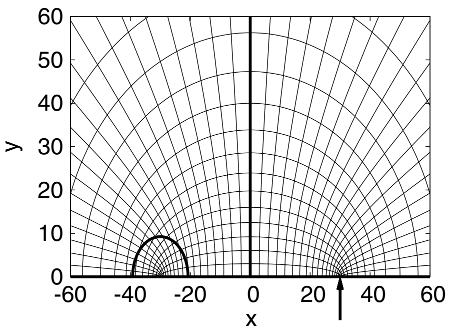


Figure 8. Example of a TwoPunctures coordinate system for BH–NS binary initial data.

and ϕ (the logarithmic conformal factor) is defined below (see (21)). In `TwoPunctures`, the conformal 3-metric $\tilde{\gamma}^{ij}$ is assumed to be flat, i.e. $\tilde{\gamma}_{ij} = \eta_{ij}$ The Bowen–York solution automatically satisfies the momentum constraint (see (100), and the Hamiltonian constraint (99) may be written as an elliptic equation for the conformal factor:

$$\Delta\psi + \tfrac{1}{8}\tilde{K}^{ij}\tilde{K}_{ij}\psi^{-7} = 0. \tag{11}$$

Decomposing the conformal factor into a singular analytical term and a regular term u, such that

$$\psi = \frac{m_1}{2r_1} + \frac{m_2}{2r_2} + u \equiv \frac{1}{\Psi} + u, \tag{12}$$

where m_1, m_2 and r_1, r_2 are the mass of and distance to each BH, respectively, and $\Psi \equiv \left(\frac{m_1}{2r_1} + \frac{m_2}{2r_2}\right)^{-1}$, the Hamiltonian constraint may be written as

$$\Delta u + \left[\tfrac{1}{8}\Psi^7\tilde{K}^{ij}\tilde{K}_{ij}\right](1 + \Psi u)^{-7} = 0 \tag{13}$$

subject to the boundary condition $u \to 1$ as $r \to \infty$. In Cartesian coordinates, the function u is infinitely differentiable everywhere except the two puncture locations. `TwoPunctures` resolves this problem by performing a coordinate transformation modeled on confocal elliptical/hyperbolic coordinates. This transforms the spatial domain shown in figure 8 into a finite cube with the object locations mapped to two parallel edges. The coordinate transformation is:

$$\begin{aligned} x &= b\frac{A^2+1}{A^2-1}\frac{2B}{1+B^2} \\ y &= b\frac{2A}{1-A^2}\frac{1-B^2}{1+B^2}\cos\phi \\ z &= b\frac{2A}{1-A^2}\frac{1-B^2}{1+B^2}\sin\phi \end{aligned} \tag{14}$$

which maps $\mathcal{R}^3$ onto $0 \leqslant A \leqslant 1$ (the elliptical quasi-radial coordinate), $-1 \leqslant B \leqslant 1$ (the hyperbolic quasi-latitudinal coordinate), and $0 \leqslant \phi < 2\pi$ (the longitudinal angle). Since u is smooth everywhere in the interior of the remapped domain, expansions into modes in these coordinates are *spectrally convergent* and thus capable of extremely high accuracy. In practice, the field is expanded into Chebyshev modes in the quasi-radial and quasi-latitudinal coordinates, and into Fourier modes around the axis connecting the two BHs. The elliptic solver uses a stabilized biconjugate gradient method to achieve rapid solutions and to avoid ill-conditioning of the spectral matrix.

5.3.4. TOVSolver. The `TOVSolver` routine in the Einstein Toolkit solves the standard TOV equations [144, 145] for the pressure, enclosed gravitational mass M_e, and gravitational potential $\Phi = \log\alpha$ in the interior of a spherically symmetric star in hydrostatic equilibrium, expressed using the Schwarzschild (areal) radius $\hat{r}$:

$$\begin{aligned}\frac{\mathrm{d}P}{\mathrm{d}\hat{r}} &= -(\mu + P)\frac{M_e + 4\pi\hat{r}^3 P}{\hat{r}(\hat{r} - 2M_e)}\\ \frac{\mathrm{d}M_e}{\mathrm{d}\hat{r}} &= 4\pi\hat{r}^2\mu\\ \frac{\mathrm{d}\Phi}{\mathrm{d}\hat{r}} &= \frac{M_e + 4\pi\hat{r}^3 P}{\hat{r}(\hat{r} - 2M_e)},\end{aligned} \tag{15}$$

where $\mu \equiv \rho(1+\epsilon)$ is the energy density of the fluid, including the internal energy contribution (this corresponds to T^{00} as measured by a comoving observer). The routine also supplies the analytically known solution in the exterior,

$$\begin{aligned}P &= P(\mathtt{TOV\_atmosphere}),\\ M_e &= M,\\ \Phi &= \frac{1}{2}\log(1 - 2M/\hat{r}),\end{aligned} \tag{16}$$

where `TOV_atmosphere` is a parameter used to specify the density of the ambient atmosphere and a polytropic EOS is assumed, and M is the total gravitational mass of the star. Since the isotropic radius r is the more commonly preferred choice to initiate dynamical calculations, the code then transforms all variables into isotropic coordinates, integrating the radius conversion formula

$$\frac{\partial(\log(r/\hat{r}))}{\partial\hat{r}} = \frac{\hat{r}^{1/2} - (\hat{r} - 2M_e)^{1/2}}{\hat{r}(\hat{r} - 2M_e)^{1/2}}, \tag{17}$$

subject to the boundary condition that in the exterior,

$$\begin{aligned}r &= \frac{1}{2}(\sqrt{\hat{r}^2 - 2M\hat{r}} + \hat{r} - M)\\ \hat{r} &= r\left(1 + \frac{M}{2r}\right)^2.\end{aligned} \tag{18}$$

In converting the solution into the variables required for a dynamical evolution, one may assume that the metric is conformally flat, with a conformal factor given by $\psi = \sqrt{\hat{r}/r}$, or equivalently, a logarithmic conformal factor $\phi = \frac{1}{2}\log(\hat{r}/r)$.

To facilitate the construction of stars in more complicated dynamical configurations, the code allows users to apply a uniform velocity to the NS, though this does not affect the ODE solution nor the resulting density profile, and thus does not represent a fully-self-consistent solution.

5.3.5. Lorene-based binary data. The Einstein Toolkit contains three routines that can read in publicly available data generated by the `Lorene` code [137, 138], though it does not currently include the capability of generating such data from scratch. For a number of reasons, such functionality is not truly required; in particular, `Lorene` is often used to generate initial data involving NSs, where the initial data problem is sufficiently hard that finding a numerical solution requires a significant amount of computer time as well as some human oversight in order to guide the routine to the correct solution. Therefore, recommended practice is to let

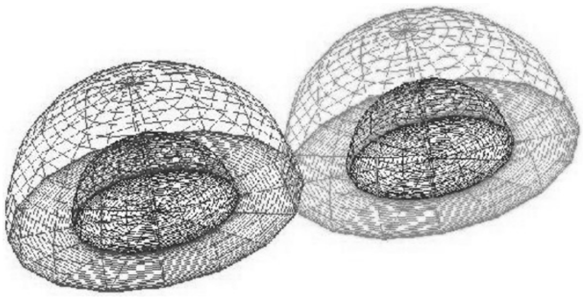


Figure 9. Example of a Lorene multi-domain coordinate system for binary initial data. The outermost, compactified domain extending to spatial infinity is not shown.

Lorene output data into files, and then read those into the Einstein Toolkit at the beginning of a run.

Lorene uses a multigrid spectral approach to solve the conformal thin-sandwich equations for binary initial configurations [139] and a single-grid spectral method for rotating stars. For binaries, five elliptic equations for the shift, lapse, and conformal factor are written down and the source terms are split into pieces that are attributed to each of the two objects. Matter source terms are ideal for this split, since they are compactly supported, while extrinsic curvature source terms are spatially extended but with sufficiently rapid falloff at large radii to yield convergent solutions. Around each object, a set of nested spheroidal sub-domains (see figure 9) is constructed to extend through all of space, with the outermost domain incorporating a compactification to allow it to extend to spatial infinity. Within each of the nested sub-domains, fields are decomposed into Chebyshev modes radially and into spherical harmonics in the angular directions, with elliptic equation solving reduced to a matrix problem. The nested sub-domains need not be perfectly spherical, and indeed one may modify the outer boundaries of each to cover any convex shape. For NSs, this allows one to map the surface of a particular sub-domain to the NS surface, minimizing Gibbs error there. For BHs, excision boundary conditions are imposed at the horizon. To read a field solution describing a given quantity onto a `Cactus`-based grid, one must incorporate the data from each star's domains at that point.

`Meudon_Bin_BH` can read in BH–BH binary initial data described in [146]. These data represent solutions to the Hamiltonian and momentum constraints, along with the trace of the spatial components of the Einstein equations, which form the linked elliptic equation set:

$$\begin{aligned}
\nabla^2\alpha_{(m)} &= \alpha\psi^4 K_{ij}K^{ij}_{(m)}\\
\nabla^2\psi_{(m)} &= -\frac{\psi^5}{8}K_{ij}K^{ij}_{(m)}\\
\nabla^2\beta^i_{(m)} + \frac{1}{3}\nabla^i\nabla_j\beta^j_{(m)} &= 2K^{ij}\left(\nabla_j\alpha_{(m)} - \frac{6\alpha}{\psi}\nabla_j\psi_{(m)}\right),
\end{aligned}$$

where the sub/superscript $(m) = 1, 2$ indexes the two BHs, and the extrinsic curvature is calculated by the Killing equation assuming helicoidal invariance, which yields the condition

$$K^{ij} = \frac{1}{2\alpha\psi^4}\left(\nabla^i\beta^j + \nabla^j\beta^i - \frac{2}{3}\eta^{ij}\nabla_k\beta^k\right). \tag{19}$$

The splitting of the extrinsic curvature into its two BH-based components is complicated, as are the boundary conditions imposed at the BH throats, and both are described in detail in

[146]; in general, though, all split quantities may be summed to reconstruct a global quantity, taking into account the asymptotic values:

$$\alpha = 1 + \alpha_{(1)} + \alpha_{(2)}; \qquad \beta^i = \beta^i_{(1)} + \beta^i_{(2)};$$
$$\psi = 1 + \psi_{(1)} + \psi_{(2)}; \qquad K^{ij} = K^{ij}_{(1)} + K^{ij}_{(2)}.$$

`Meudon_Bin_NS` handles binary NS data described in [138], which represent solutions of the equations

$$\nabla^2 \nu_{(m)} = 4\pi\psi^4(E_{(m)} + S_{(m)}) + \psi^4 K_{ij}K^{ij}_{(m)} - \nabla_i \nu_{(m)} \nabla^i \beta$$
$$\nabla^2 \beta_{(m)} = 4\pi\psi^4 S_{(m)} + \tfrac{3}{4}\psi^4 K_{ij}K^{ij}_{(m)} - \tfrac{1}{2}(\nabla_i \nu_{(m)} \nabla^i \nu + \nabla_i \beta_{(m)} \nabla^i \beta)$$
$$\nabla^2 \beta^i_{(m)} + \tfrac{1}{3}\nabla^i \nabla_j \beta^j_{(m)} = -16\pi\alpha\psi^4(E_{(m)} + P_{(m)})v^i_{(m)} + 2\alpha\psi^4 K^{ij}_{(m)} \nabla_j(3\beta - 4\nu),$$

where ν and β are defined as

$$\nu \equiv \log\alpha; \qquad \beta \equiv \ln \alpha\psi^2. \tag{20}$$

These equations are merely convenient reparameterizations of the ones used to generate binary BH data, with the matter source terms included. The extrinsic curvature is computed using (19), with both K^{ij} and β^i replaced by the split versions. The matter sources terms E and S, representing projections of the stress–energy tensor, are defined in (34) and (36) below. Lorene allows for two different NS spin states, either irrotational or synchronized. In the synchronized case, the velocity may be specified as a function of position once the orbital velocity is determined, while the irrotational case yields a rather complicated differential equation for the velocity potential which may then be used to determine the corresponding 3-velocity (see equation (38) of [138]). `Meudon_Mag_NS` may be used to read in magnetized isolated NS data [137, 147]. The spacetime metric is determined using the same equations as for a binary NS configuration, but with magnetic contributions added to the matter source terms via their contributions to the stress–energy tensor, and no splitting required. The magnetic field 4-vector, which is assumed only to have non-zero components in the t and ϕ directions, may be determined from two Poisson-type equations.

5.4. Spacetime curvature evolution

The Einstein Toolkit curvature evolution code `McLachlan` [148, 118] is auto-generated from tensor equations via `Kranc` (section 4.4). It implements the Einstein equations in a 3 + 1 split as a Cauchy initial boundary value problem [130]. For this, the BSSN conformal-tracefree reformulation [9, 10, 149] of the original ADM formalism [129] is employed. `McLachlan` uses finite differencing for the spacetime variables (orders 2, 4, 6 and 8 are currently implemented) and adds a Kreiss–Oliger dissipation term to remove high-frequency noise. The evolved variables are the logarithmic conformal factor ϕ (the W method [150] is also implemented), the conformal 3-metric $\tilde{\gamma}_{ij}$, the trace K of the extrinsic curvature, the conformal trace free extrinsic curvature $\tilde{A}_{ij}$ and the conformal connection functions $\tilde{\Gamma}^i$. These are defined in terms of the standard ADM 4-metric g_{ij}, 3-metric γ_{ij}, and extrinsic curvature K_{ij} by:

$$\phi \equiv \log\left[\tfrac{1}{12}\det\gamma_{ij}\right], \tag{21}$$

$$\tilde{\gamma}_{ij} \equiv \mathrm{e}^{-4\phi}\gamma_{ij}, \tag{22}$$

$$K \equiv g^{ij}K_{ij}, \tag{23}$$

$$\tilde{A}_{ij} \equiv \mathrm{e}^{-4\phi}\left[K_{ij} - \tfrac{1}{3}g_{ij}K\right], \tag{24}$$

$$\tilde{\Gamma}^i \equiv \tilde{\gamma}^{jk}\tilde{\Gamma}^i_{jk}. \tag{25}$$

The evolution equations are then:

$$\partial_0\alpha = -\alpha^2 f(\alpha,\phi,x^\mu)(K - K_0(x^\mu)) \tag{26}$$

$$\partial_0 K = -\gamma^{ij}\tilde{D}_i\tilde{D}_j\alpha + \alpha\left(\tilde{A}^{ij}\tilde{A}_{ij} + \tfrac{1}{3}K^2\right) + 4\pi(E + \gamma^{ij}S_{ij}) \tag{27}$$

$$\partial_0\beta^i = \alpha^2 G(\alpha,\phi,x^\mu)B^i \tag{28}$$

$$\partial_0 B^i = \mathrm{e}^{-4\phi}H(\alpha,\phi,x^\mu)\partial_0\tilde{\Gamma}^i - \eta^i(B^i,\alpha,x^\mu) \tag{29}$$

$$\partial_0\phi = -\tfrac{1}{6}(\alpha K - \partial_k\beta^k) \tag{30}$$

$$\partial_0\tilde{\gamma}_{ij} = -2\alpha\tilde{A}_{ij} + 2\tilde{\gamma}_{k(i}\partial_{j)}\beta^k - \tfrac{2}{3}\tilde{\gamma}_{ij}\partial_k\beta^k \tag{31}$$

$$\begin{aligned}\partial_0\tilde{A}_{ij} = {}& \mathrm{e}^{-4\phi}\left[\alpha\tilde{R}_{ij} + \alpha R^\phi_{ij} - \tilde{D}_i\tilde{D}_j\alpha\right]^{TF} \\ & + \alpha K\tilde{A}_{ij} - 2\alpha\tilde{A}_{ik}\tilde{A}^k{}_j + 2\tilde{A}_{k(i}\partial_{j)}\beta^k - \tfrac{2}{3}\tilde{A}_{ij}\partial_k\beta^k - 8\pi\alpha\,\mathrm{e}^{-4\phi}S^{TF}_{ij}\end{aligned} \tag{32}$$

$$\begin{aligned}\partial_0\tilde{\Gamma}^i = {}& -2\tilde{A}^{ij}\partial_j\alpha + 2\alpha\left[\tilde{\Gamma}^i{}_{kl}\tilde{A}^{kl} + 6\tilde{A}^{ij}\partial_j\phi - \tfrac{2}{3}\tilde{\gamma}^{ij}K_{,j}\right] \\ & - \tilde{\Gamma}^j\partial_j\beta^i + \tfrac{2}{3}\tilde{\Gamma}^i\partial_j\beta^j + \tfrac{1}{3}\tilde{\gamma}^{ik}\beta^j{}_{,jk} + \tilde{\gamma}^{jk}\beta^i{}_{,jk} - 16\pi\alpha\tilde{\gamma}^{ik}S_k.\end{aligned} \tag{33}$$

The stress–energy tensor $T_{\mu\nu}$ is incorporated via the projections

$$E \equiv n_\alpha n_\beta T^{\alpha\beta} = \frac{1}{\alpha^2}(T_{00} - 2\beta^i T_{0i} + \beta^i\beta^j T^{ij}) \tag{34}$$

$$S_{ij} \equiv \gamma_{i\alpha}\gamma_{j\beta}T^{\alpha\beta} \tag{35}$$

$$S \equiv S^i_i = \gamma^{ij}S_{ij} \tag{36}$$

$$S_i \equiv -\gamma_{i\alpha}n_\beta T^{\alpha\beta} = -\frac{1}{\alpha}(T_{0i} - \beta^j T_{ij}). \tag{37}$$

We have introduced the notation $\partial_0 = \partial_t - \beta^j\partial_j$. All quantities with a tilde involve the conformal 3-metric $\tilde{\gamma}_{ij}$, which is used to raise and lower indices. In particular, $\tilde{D}_i$ and $\tilde{\Gamma}^k_{ij}$ refer to the covariant derivative and the Christoffel symbols with respect to $\tilde{\gamma}_{ij}$. The expression $[\cdots]^{TF}$ denotes the trace-free part of the expression inside the parentheses, and we define the Ricci tensor contributions as:

$$\tilde{R}_{ij} = -\tfrac{1}{2}\tilde{\gamma}^{kl}\partial_k\partial_l\tilde{\gamma}_{ij} + \tilde{\gamma}_{k(i}\partial_{j)}\tilde{\Gamma}^k - \partial_k\tilde{\gamma}_{l(i}\partial_{j)}\tilde{\gamma}^{kl} + \tfrac{1}{2}\tilde{\Gamma}^l\tilde{\gamma}_{ij,l} - \tilde{\Gamma}^l_{ik}\tilde{\Gamma}^k_{jl} \tag{38}$$

$$R^\phi_{ij} = -2\tilde{D}_i\tilde{D}_j\phi - 2\tilde{\gamma}_{ij}\tilde{D}^k\tilde{D}_k\phi + 4\tilde{D}_i\phi\,\tilde{D}_j\phi - 4\tilde{\gamma}_{ij}\tilde{D}^k\phi\,\tilde{D}_k\phi. \tag{39}$$

This is a so-called ϕ-variant of BSSN. The evolved gauge variables are lapse α, shift β^i, and a quantity B^i related to the time derivative of the shift. The gauge functions f, K_0, G, H, and η are determined by our choice of a $1 + \log$ [151] slicing:

$$f(\alpha,\phi,x^\mu) \equiv 2/\alpha \tag{40}$$

$$K_0(x^\mu) \equiv 0 \tag{41}$$

and Γ-driver shift condition [151]:

$$G(\alpha,\phi,x^\mu) \equiv (3/4)\,\alpha^{-2} \tag{42}$$

$$H(\alpha,\phi,x^\mu) \equiv \mathrm{e}^{4\phi} \tag{43}$$

$$\eta(B^i, \alpha, x^\mu) \equiv (1/2)\, B^i q(r). \tag{44}$$

The expression $q(r)$ attenuates the Γ-driver depending on the radius as described below.

The Γ-driver shift condition is symmetry-seeking, driving the shift β^i to a state that renders the conformal connection functions $\tilde{\Gamma}^i$ stationary. Of course, such a stationary state cannot be achieved while the metric is evolving, but in a stationary spacetime the time evolution of the shift β^i and thus that of the spatial coordinates x^i will be exponentially damped. This damping timescale is set by the gauge parameter η (see (44)) which has dimension $1/T$ (inverse time). As described in [152, 153], this timescale may need to be adapted in different regions of the domain to avoid spurious high-frequency behavior in regions that otherwise evolve only very slowly, e.g., far away from the source.

Here we use the simple damping mechanism described in (12) of [153], through defining the function q in (44):

$$q(r) \equiv \begin{cases} 1 & \text{for} \quad r \leqslant R \quad \text{(near the origin)} \\ R/r & \text{for} \quad r \geqslant R \quad \text{(far away)} \end{cases} \tag{45}$$

with a constant R defining the transition radius between the interior, where $q \approx 1$, and the exterior, where q falls off as $1/r$.

5.4.1. Initial conditions. Initial conditions for the ADM variables g_{ij}, K_{ij}, lapse α, and shift β^i are provided by the initial data routines discussed in section 5.3. From these the BSSN quantities are calculated via their definitions, setting $B^i = 0$, and using cubic extrapolation for $\tilde{\Gamma}^i$ at the outer boundary. This extrapolation is necessary since the $\tilde{\Gamma}^i$ are calculated from spatial derivatives of the metric, and one cannot use centered finite differencing stencils near the outer boundary.

The extrapolation stencils distinguish between points on the faces, edges, and corners of the grid. Points on the faces are extrapolated via stencils perpendicular to that face, while points on the edges and corners are extrapolated with stencils aligned with the average of the normals of the adjoining faces. For example, points on the $(+x, +y)$ edge are extrapolated in the $(1, 1, 0)$ direction, while points in the $(+x, +y+z)$ corner are extrapolated in the $(1, 1, 1)$ direction. Since several layers of boundary points have to be filled for higher order schemes (e.g., three layers for a fourth-order scheme), one proceeds outward starting from the innermost layer. Each subsequent layer is then defined via the points in the interior and the previously calculated layers.

5.4.2. Boundary conditions. During time evolution, a Sommerfeld-type radiative boundary condition is applied to all components of the evolved BSSN variables as described in [149]. The main feature of this boundary condition is that it assumes approximate spherical symmetry of the solution, while applying the actual boundary condition on the boundary of a cubic grid where the face normals are not aligned with the radial direction. This boundary condition defines the RHS of the BSSN state vector on the outer boundary, which is then integrated in time as well so that the boundary and interior are calculated with the same order of accuracy.

The main part of the boundary condition assumes that one has an outgoing radial wave with some speed v_0:

$$X = X_0 + \frac{u(r - v_0 t)}{r}, \tag{46}$$

where X is any of the tensor components of evolved variables, X_0 the value at infinity, and u a spherically symmetric perturbation. Both X_0 and v_0 depend on the particular variable and have to be specified. This implies the following differential equation:

$$\partial_t X = -v^i \partial_i X - v_0 \frac{X - X_0}{r}, \tag{47}$$

where $v^i = v_0 x^i / r$. The spatial derivatives ∂_i are evaluated using centered finite differencing where possible, and one-sided finite differencing elsewhere. Second-order stencils are used in the current implementation.

In addition to this main part, it is also necessary to account for those parts of the solution that do not behave as a pure wave, e.g., Coulomb type terms caused by infall of the coordinate lines. It is assumed that these parts decay with a certain power p of the radius. This is implemented by considering the radial derivative of the source term above, and extrapolating according to this power-law decay.

Given a source term $(\partial_t X)$, one defines the corrected source term $(\partial_t X)^*$ via

$$(\partial_t X)^* = (\partial_t X) + \left(\frac{r}{r - n^i \partial_i r}\right)^p n^i \partial_i (\partial_t X), \tag{48}$$

where n^i is the normal vector of the corresponding boundary face. The spatial derivatives ∂_i are evaluated by comparing neighboring grid points, corresponding to a second-order stencil evaluated in the middle between the two neighboring grid points. Although strictly speaking different variables or even different components of tensors can fall off differently, we find that using $p = 2$ works well in practice.

As with the initial conditions above, this boundary condition is evaluated on several layers of grid points, starting from the innermost layer. Both the extrapolation and radiative boundary condition algorithms are implemented in the publicly available `NewRad` component of the Einstein Toolkit.

This boundary condition is only a coarse approximation of the actual decay behavior of the BSSN state vector, and it does not capture the correct behavior of the evolved variables. However, one finds that this boundary condition leads to stable evolutions if applied sufficiently far from the source. Errors introduced at the boundary (both errors in the geometry and constraint violations) propagate inward with the speed of light [148]. Gauge changes introduced by the boundary condition, which are physically not observable, propagate faster, with a speed up to $\sqrt{2}$ for the gauge conditions used in `McLachlan`.

Unfortunately more advanced radiative and/or constraint preserving boundary conditions are not available in the Einstein Toolkit at the present time. However, `Cactus` is a flexible and extensible framework and supports data structures (such as lower dimensional grid arrays) that could be used for implementing more advanced outer boundary conditions. It will be the requirements and needs of the numerical relativity community that decides which enhancements and improvements will be added to the toolkit.

5.5. Hydrodynamics evolution

Hydrodynamic evolution in the Einstein Toolkit is designed so that it interacts with the metric curvature evolution through a small set of variables, allowing for maximum modularity in implementing, editing, or replacing either evolution scheme.

The primary hydrodynamics evolution routine in the Einstein Toolkit is `GRHydro`, a code derived from the public `Whisky` code [100, 154–156] designed primarily by researchers at AEI and their collaborators. It includes a high resolution shock-capturing (HRSC) scheme to evolve hydrodynamic quantities, with several different reconstruction methods and Riemann solvers, as we discuss below. In such a scheme, we define a set of ‘conserved’ hydrodynamic variables, defined in terms of the ‘primitive’ physical variables such as mass density, specific internal energy, pressure, and velocity. Wherever derivatives of hydrodynamic terms appear in the evolution equations for the conserved variables, they are restricted to appear only inside divergence terms (referred to as fluxes) and never in the source terms. By calculating fluxes between pairs of neighboring points where field and hydrodynamic terms are evaluated, we obtain a consistent description of time evolution using HRSC reconstruction techniques that accounts for the fact that hydrodynamic variables are not smooth and may not be finite differenced accurately[13]. All other source terms in the evolution equations may contain only the hydrodynamic variables themselves and the metric variables and derivatives of the latter, since the metric must formally be smooth and thus differentiable using finite differencing techniques. Summarizing these methods briefly, the following stages occur every time step:

- The primitive variables are 'reconstructed' between neighboring points using shock-capturing techniques, with total variation diminishing (TVD), piecewise parabolic (PPM), and essentially non-oscillatory (ENO) methods currently implemented.
- A Riemann problem is solved to generate the fluxes corresponding to the reconstructed values using an approximate solver. Currently implemented versions include Harten–Lax–van Leer–Einfeldt (HLLE), Roe, and Marquina solvers.
- Using the computed flux and source terms, the conserved variables are advanced one time step, and used to recalculate the new values of the primitive variables.

We discuss the GRHD formalism, the stages within a time step, and the other aspects of the code below, noting that the documentation included in the released version is quite extensive and covers many of these topics in substantially more detail.

5.5.1. Ideal GRHD. The equations of ideal GR hydrodynamics [132] evolved by `GRHydro` are derived from the local GR conservation laws of mass and energy-momentum:

$$\nabla_\mu J^\mu = 0, \qquad \nabla_\mu T^{\mu\nu} = 0, \tag{49}$$

where ∇_μ denotes the covariant derivative with respect to the 4-metric, and $J^\mu = \rho u^\mu$ is the mass current. $T^{\mu\nu}$ is the stress–energy tensor, defined in (6) as $T^{\mu\nu} = \rho h u^\mu u^\nu + P g^{\mu\nu}$, where u^μ is the 4-velocity and ρ the rest-mass density.

The 3-velocity v^i may be calculated in the form

$$v^i = \frac{u^i}{W} + \frac{\beta^i}{\alpha}, \tag{50}$$

where

$$W \equiv \alpha u^0 = (1 - v^i v_i)^{-1/2} \tag{51}$$

is the Lorentz factor. The contravariant 4-velocity is then given by:

$$u^0 = \frac{W}{\alpha}, \qquad u^i = W\left(v^i - \frac{\beta^i}{\alpha}\right), \tag{52}$$

and the covariant 4-velocity is:

$$u_0 = W(v^i\beta_i - \alpha), \qquad u_i = W v_i. \tag{53}$$

The `GRHydro` evolution scheme is a first-order hyperbolic flux-conservative system for the conserved variables D, S_i, and τ, which may be defined in terms of the primitive variables ρ, ϵ, v_i, such that:

$$D = \sqrt{\gamma}\rho W, \tag{54}$$

[13] While the `Carpet` AMR driver uses a so-called vertex-centered approach, `GRHydro` is sufficiently general in its techniques that it may be used with either 'cell-centered' or 'vertex-centered' grid infrastructures; see, e.g., [128] for a review of these methods.

$$S_i = \sqrt{\gamma}\rho h W^2 v_i, \tag{55}$$

$$\tau = \sqrt{\gamma}(\rho h W^2 - P) - D, \tag{56}$$

where γ is the determinant of γ_{ij}. The evolution system then becomes

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}^i}{\partial x^i} = \mathbf{S}, \tag{57}$$

with

$$\begin{aligned} \mathbf{U} &= [D, S_j, \tau], \\ \mathbf{F}^i &= \alpha\left[D\tilde{v}^i, S_j\tilde{v}^i + \delta^i_j P, \tau\tilde{v}^i + Pv^i\right], \\ \mathbf{S} &= \alpha\left[0, T^{\mu\nu}\left(\frac{\partial g_{\nu j}}{\partial x^\mu} - \Gamma^\lambda_{\mu\nu} g_{\lambda j}\right), \alpha\left(T^{\mu 0}\frac{\partial \ln\alpha}{\partial x^\mu} - T^{\mu\nu}\Gamma^0_{\mu\nu}\right)\right]. \end{aligned} \tag{58}$$

Here, $\tilde{v}^i = v^i - \beta^i/\alpha$ and $\Gamma^\lambda_{\mu\nu}$ are the 4-Christoffel symbols. The time integration and coupling with curvature are carried out with the MoL [157]. The expressions for $\mathbf{S}$ are calculated in `GRHydro` by using the definition of the extrinsic curvature to avoid any time derivatives whatsoever, as discussed in detail in the code's documentation, following a suggestion by Mark Miller based on experience with the `GR3D` code.

5.5.2. Reconstruction techniques. In order to calculate fluxes at cell faces, we first must calculate function values on either side of the face. In practice, reconstructing the primitive variables yields more stable and accurate evolutions than reconstructing the conservatives. In what follows, we assume a Cartesian grid and loop over faces along each direction in turn. We define $q^L_{i+1/2}$ to be the value of a quantity q on the left side of the face between $q_i \equiv q(x_i, y, z)$ and $q_{i+1} \equiv q(x_{i+1}, y, z)$, where x_i is the ith point in the x-direction, and $q^R_{i+1/2}$ the right side of the same face.

For TVD methods, we let:

$$q^L_{i+1/2} = q_i + \frac{f(q_i)\Delta x}{2}; \qquad q^R_{i+1/2} = q_{i+1} - \frac{f(q_{i+1})\Delta x}{2}, \tag{59}$$

where $f(q_i)$ is a slope-limited gradient function, typically determined by the values of $q_{i+1} - q_i$ and $q_i - q_{i-1}$, with a variety of different forms of the slope limiter available. In practice, all try to accomplish the same task of preserving monotonicity and removing the possibility of spuriously creating local extrema. Implemented methods include minmod, superbee (both [158]), and monotonized central [159].

The PPM is a multi-step method based around a quadratic fit to nearby points interpolated to cell faces [160], for which q^L and q^R are generally equivalent except near shocks and local extrema. The version implemented in `GRHydro` includes the steepening and flattening routines described in the original PPM papers, with a simplified dissipation procedure. ENO methods use a divided differences approach to achieve third-order accuracy via polynomial interpolation [161, 162].

Both ENO and PPM yield third-order accuracy for smooth monotonic functions, whereas TVD methods typically yield second-order accurate values. Regardless of the reconstruction scheme chosen, all of these methods reduce to first order near local extrema and shocks.

5.5.3. Riemann solvers. The Riemann problem involves the solution of the equation

$$\partial_t q + \partial_i f^i(q) = 0 \tag{60}$$

at some point X representing a discontinuity between constant states. The exact solution can be quite complicated, involving five different waves with different characteristic speeds for a hydrodynamic problem (8 for GRMHD), so `GRHydro` implements three different approximate solvers to promote computational efficiency. In each case, the solution takes a self-similar form $q(\xi)$, where $\xi \equiv (x - X)/t$ represents the characteristic speed from the original shock location to the point in question in space and time.

The simplest method implemented is the HLLE solver [163, 164] (HLL or HLLE, depending on the reference), which uses a two wave approximation to calculate the evolution along the shock front. With ξ_- and ξ_+ the most negative and most positive wave speeds present on either side of the interface, the solution $q(\xi)$ is assumed to take the form

$$q = \begin{cases} q^L & \text{if} \quad \xi < \xi_- \\ q_* & \text{if} \quad \xi_- < \xi < \xi_+ \\ q^R & \text{if} \quad \xi > \xi_+, \end{cases} \tag{61}$$

with the intermediate state q_* given by

$$q_* = \frac{\xi_+ q^R - \xi_- q^L - f(q^R) + f(q^L)}{\xi_+ - \xi_-}. \tag{62}$$

The numerical flux along the interface takes the form

$$f(q) = \frac{\widehat{\xi}_+ f(q^L) - \widehat{\xi}_- f(q^R) + \widehat{\xi}_+ \widehat{\xi}_- (q^R - q^L)}{\widehat{\xi}_+ - \widehat{\xi}_-}, \tag{63}$$

where

$$\widehat{\xi}_- = \min(0, \xi_-), \qquad \widehat{\xi}_+ = \max(0, \xi_+). \tag{64}$$

It is these flux terms that are then used to evolve the hydrodynamic quantities.

The Roe solver [165] involves linearizing the evolution system for the hydrodynamic evolution, defining the Jacobian matrix $A \equiv \frac{\partial f}{\partial q}$ (see (60)), and working out the eigenvalues λ^i and left and right eigenvectors, l_i and r^j, assumed to be orthonormalized so that $l_i \cdot r^j = \delta_i^j$. Defining the characteristic variables $w_i = l_i \cdot q$, the characteristic equation becomes

$$\partial_t w + \Lambda \partial_x w = 0 \tag{65}$$

with Λ the diagonal matrix of eigenvalues. Letting $\Delta w_i \equiv w_i^L - w_i^R = l_i \cdot (q^L - q^R)$ represent the differences in the characteristic variables across the interface, the Roe flux is calculated as

$$f(q) = \frac{1}{2}\left(f(q^L) + f(q^R) - \sum |\lambda_i| \Delta w_i r^i \right), \tag{66}$$

where the eigenvectors appearing in the summed term are evaluated for the approximate Roe average flux $q_{\text{Roe}} = \frac{1}{2}(q^L + q^R)$. The Marquina flux routines use a similar approach to the Roe method, but provide a more accurate treatment for supersonic flows (i.e. those for which the characteristic wave with $\xi = 0$ is within a rarefaction zone) [166, 167].

5.5.4. Conservative-to-primitive conversion. In order to invert equations (54)–(56), solving for the primitive variables based on the values of the conservative ones, `GRHydro` uses a one-dimensional Newton–Raphson approach that solves for a consistent value of the pressure. Defining the (known) undensitized conservative variables $\hat{D} \equiv D/\sqrt{\gamma} = \rho W$, $\hat{S}_i = S_i/\sqrt{\gamma} = \rho h W^2 v_i$ and $\hat{\tau} \equiv \tau/\sqrt{\gamma} = \rho h W^2 - P - \hat{D}$, as well as the auxiliary quantities $Q \equiv \rho h W^2 = \hat{\tau} + \hat{D} + P$ and $\hat{S}^2 = \gamma_{ij}\hat{S}^i\hat{S}^j = (\rho h W)^2(W^2 - 1)$, the former of which depends on P and the latter of which is known, we find that $\sqrt{Q^2 - \hat{S}^2} = \rho h W$ and thus

$$\rho = \frac{\hat{D}\sqrt{Q^2 - \hat{S}^2}}{Q} \tag{67}$$

$$W = \frac{Q}{\sqrt{Q^2 - \hat{S}^2}} \tag{68}$$

$$\epsilon = \frac{\sqrt{Q^2 - \hat{S}^2} - PW - \hat{D}}{\hat{D}}. \tag{69}$$

Given the new values of ρ and ϵ, one may then find the residual between the pressure and $P(\rho, \epsilon)$ and perform the Newton–Raphson step, so long as the values of $\frac{\partial P}{\partial \rho}$ and $\frac{\partial P}{\partial \epsilon}$ are known.

5.5.5. Atmospheres, boundaries, and other code details. `GRHydro` uses an atmosphere, or extremely-low density floor, to avoid problems involving sound speeds and conservative-to-primitive variable conversion near the edges of matter distributions. The floor density value may be chosen in either absolute (`rho_abs_min`) or relative (`rho_rel_min`) terms. The atmosphere is generally assumed to have a specified polytropic EOS, regardless of the EOS describing the rest of the matter within the simulation. Whenever the numerical evolution results in a grid cell where conservative-to-primitive variable conversion yields negative values of either ρ or ϵ, the cell is reassigned to the atmosphere, with 0-velocity.

At present, only flat boundary conditions are supported for hydrodynamic variables, since it is generally recommended that the outer boundaries of the simulation be placed far enough away so that all cells near the edge of the computational domain represent the atmosphere. `GRHydro` has the ability to advect a set of passive scalars, referred to as 'tracers', as well as the electron fraction of a fluid, under the assumption that each tracer X follows the conservation law

$$\partial_t(DX) + \partial_i(\alpha \tilde{v}^i DX) = 0. \tag{70}$$

5.6. Equations of state

An EOS connecting the primitive state variables is needed to close the system of GR hydrodynamics equations. The module `EOS_Omni` provides a unified general EOS interface and back-end for simple analytic and complex microphysical EOSs.

The polytropic EOS

$$P = K\rho^\Gamma, \tag{71}$$

where K is the polytropic constant and Γ is the adiabatic index, is appropriate for adiabatic (= isentropic) evolution without shocks. When using the polytropic EOS, one does not need to evolve the total fluid energy equation, since the specific internal energy ϵ is fixed to

$$\epsilon = \frac{K\rho^\Gamma}{(\Gamma - 1)\rho}. \tag{72}$$

Note that the adiabatic index $\Gamma = \mathrm{d}\ln P/\mathrm{d}\ln\rho$ is related to the frequently used polytropic index n via $n = 1/(\Gamma - 1)$.

The gamma-law EOS[14],

$$P = (\Gamma - 1)\rho\epsilon, \tag{73}$$

allows for non-adiabatic flow but still assumes fixed microphysics, which is encapsulated in the constant adiabatic index Γ. This EOS has been used extensively in simulations of NS–NS and BH–NS mergers.

[14] For historical reasons, this EOS is referred to as the 'ideal fluid' EOS in `GRHydro`.

The hybrid EOS, first introduced by [168], is a 2-piecewise polytrope with a thermal component designed for the application in simple models of stellar collapse. At densities below nuclear density, a polytropic EOS with $\Gamma = \Gamma_1 \approx 4/3$ is used. To mimic the stiffening of the nuclear EOS at nuclear density, the low-density polytrope is matched to a second polytrope with $\Gamma = \Gamma_2 \gtrsim 2$. To allow for thermal contributions to the pressure due to shock heating, a gamma-law with $\Gamma = \Gamma_{\rm th}$ is used. The full EOS then reads

$$P = \frac{\Gamma - \Gamma_{\rm th}}{\Gamma - 1} K \rho_{\rm nuc}^{\Gamma_1 - \Gamma} \rho^{\Gamma} - \frac{(\Gamma_{\rm th} - 1)(\Gamma - \Gamma_1)}{(\Gamma_1 - 1)(\Gamma_2 - 1)} K \rho_{\rm nuc}^{\Gamma_1 - 1} \rho + (\Gamma_{\rm th} - 1)\rho\epsilon\,. \quad (74)$$

In this, the total specific internal energy ϵ consists of a polytropic and a thermal contribution. The variable Γ assumes one of two values,

$$\Gamma = \begin{cases} \Gamma_1 & \text{for} \quad \rho < \rho_{\rm nuc}, \\ \Gamma_2 & \text{for} \quad \rho \geqslant \rho_{\rm nuc}, \end{cases} \quad (75)$$

where $\rho_{\rm nuc}$ is typically set to $2 \times 10^{14}\,{\rm g\,cm^{-3}}$.

In iron core collapse, the pressure below nuclear density is dominated by the pressure of relativistically degenerate electrons. For this, one sets $K = 4.897 \times 10^{14}$ (cgs) in the above. The thermal index $\Gamma_{\rm th}$ is usually set to 1.5, corresponding to a mixture of relativistic ($\Gamma = 4/3$) and non-relativistic ($\Gamma = 5/3$) gas. Provided appropriate choices of EOS parameters (e.g., [169]), the hybrid EOS leads to qualitatively correct collapse and bounce dynamics in stellar collapse.

EOS_Omni also integrates the nuc_eos driver routine, which was first developed for the GR1D code [55] for tabulated microphysical finite-temperature EOS which assume nuclear statistical equilibrium. nuc_eos handles EOS tables in HDF5 format which contain entries for thermodynamic variables $X = X(\rho, T, Y_e)$, where T is the matter temperature and Y_e is the electron fraction. nuc_eos also supports calls for $X = X(\rho, \epsilon, Y_e)$ and carries out a Newton iteration to find $T(\rho, \epsilon, Y_e)$. For performance reasons, nuc_eos employs simple tri-linear interpolation in thermodynamic quantities and thus requires finely spaced tables to maintain thermodynamic consistency at an acceptable level. EOS tables in the format required by nuc_eos are freely available from http://stellarcollapse.org.

5.7. Analysis

It is often beneficial and sometimes necessary to evaluate analysis quantities during the simulation rather than post-processing variable output. Beyond extracting physics, these quantities are often used as measures of how accurately the simulation is progressing. In the following, we describe the common quantities available through Einstein Toolkit modules, and how different modules approach these quantities with differing assumptions and algorithms. The most common analysis quantities provided fall broadly into several different categories, including horizons, masses and momenta, and gravitational waves. Note that several modules bridge these categories and some fall outside them, including routines to perform constraint monitoring and to compute commonly used derived spacetime quantities. The following discussion is meant as an overview of the most common tools rather than an exhaustive list of the functionality provided by the Einstein Toolkit. In most cases, the analysis modules work on the variables stored in the base modules discussed in section 5.2, ADMBase, TmunuBase, and HydroBase, to be as portable as possible.

5.7.1. Horizons. When spacetimes contain a BH, localizing its horizon is necessary for describing time-dependent quasi-local measures such as its mass and spin. The Einstein Toolkit provides two modules—AHFinder and AHFinderDirect—for locating the AHs defined locally on a hypersurface. The module `EHFinder` is also available to search an evolved spacetime for the globally defined event horizons.

`EHFinder` [170] evolves a null surface backward in time given an initial guess (e.g., the last AH) which will, in the vicinity of an event horizon, converge exponentially to its location. This must be done after a simulation has already evolved the initial data forward in time with enough 3D data written out that the full 4-metric can be recovered at each time step.

In `EHFinder`, the null surface is represented by a function $f(t, x^i) = 0$ which is required to satisfy the null condition $g^{\alpha\beta}(\partial_\alpha f)(\partial_\beta f) = 0$. In the standard numerical 3+1 form of the metric, this null condition can be expanded out into an evolution equation for f as

$$\partial_t f = \beta^i \partial_i f - \sqrt{\alpha^2 \gamma^{ij}(\partial_i f)(\partial_j f)}, \tag{76}$$

where the roots are chosen to describe outgoing null geodesics. The function f is chosen such that it is negative within the initial guess of the horizon and positive outside, initially set to a distance measure from the initial surface guess $f(t_0, x^i) = \sqrt{\left(x^i - x_0^i\right)\left(x_i - x_{i(0)}\right)} - r_0$. There is a numerical problem with the steepening of ∇f during the evolution, so the function f is regularly re-initialized during the evolution to satisfy $|\nabla f| \simeq 1$. This is done by evolving

$$\frac{\mathrm{d}f}{\mathrm{d}\lambda} = -\frac{f}{\sqrt{f^2+1}}\left(|\nabla f| - 1\right) \tag{77}$$

for some unphysical parameter λ until a steady state has been reached. As the isosurface $f = 0$ converges exponentially to the event horizon, it is useful to evolve two such null surfaces which bracket the approximate position of the anticipated event horizon to further narrow the region containing the event horizon.

However, event horizons can only be found after the full spacetime has been evolved. It is often useful to know the positions and shapes of any BH on a given hypersurface for purposes such as excision, accretion, and local measures of its mass and spin. The Einstein Toolkit provides several algorithms of varying speed and accuracy to find marginally trapped surfaces, of which the outermost are apparent horizons (AHs). All finders make use of the fact that null geodesics have vanishing expansion on an AH which, in the usual 3+1 quantities, can be written

$$\Theta \equiv \nabla_i n^i + K_{ij} n^i n^j - K = 0, \tag{78}$$

where n^i is the unit outgoing normal to the 2-surface.

The module `AHFinder` provides two algorithms for locating AHs. The minimization algorithm [171] finds a local surface S with a minimal value for $\oint_S (\Theta - \Theta_o)^2 \mathrm{d}^2 S$ corresponding to a surface of constant expansion Θ_o, with $\Theta_o = 0$ corresponding to the AH. For time-symmetric data, the option exists to find instead the minimum of the surface area, which in this case corresponds to an AH. An alternative algorithm is provided by `AHFinder`, the flow algorithm [172], on which `EHFinder` is also based. Defining a surface as a level set $f(x^i) = r - h(\theta, \phi) = 0$, and introducing an unphysical timelike parameter λ to parametrize the flow of h toward a solution, (78) can be rewritten

$$\partial_\lambda h = -\left(\frac{\alpha}{\ell_{\max}(\ell_{\max}+1)} + \beta\right)\left(1 - \frac{\beta}{\alpha} L^2\right)^{-1} \rho\Theta, \tag{79}$$

where ρ is a strictly positive weight, L^2 is the Laplacian of the 2D metric, and α, β, and $\ell_{\max}$ are free parameters. Decomposing $h(\theta, \phi)$ onto a basis of spherical harmonics, the coefficients $a_{\ell m}$ evolve iteratively toward a solution as

$$a_{\ell m}^{(n+1)} = a_{\ell m}^{(n)} - \frac{\alpha + \beta\ell_{\max}(\ell_{\max}+1)}{\ell_{\max}(\ell_{\max}+1)(1 + \beta\ell(\ell+1)/\alpha)} (\rho\Theta)_{\ell m}^{(n)}. \tag{80}$$

The `AHFinderDirect` module [173] is a faster alternative to `AHFinder`. Its approach is to view (78) as an elliptic PDE for $h(\theta, \phi)$ on S^2 using standard finite differencing methods. Rewriting (78) in the form

$$\Theta \equiv \Theta(h, \partial_u h, \partial_{uv} h; \gamma_{ij}, K_{ij}, \partial_k \gamma_{ij}) = 0, \tag{81}$$

the expansion Θ is evaluated on a trial surface, then iterated using a Newton–Raphson method to solve $\mathbf{J} \cdot \delta h = -\Theta$, where $\mathbf{J}$ is the Jacobian matrix. The drawback of this method is that it is not guaranteed to give the outermost marginally trapped surface. In practice however, this limitation can be easily overcome by either a single good initial guess, or multiple less accurate initial guesses.

5.7.2. Masses and momenta. Two distinct measures of mass and momenta are available in relativistic spacetimes. First, ADM mass and angular momentum evaluated as either surface integrals at infinity or volume integrals over entire hypersurfaces give a measure of the total energy and angular momentum in the spacetime. The module `ML_ADMQuantities` of the `McLachlan` code [174] uses the latter method, creating gridfunctions containing the integrand of the volume integrals [175]:

$$M = \frac{1}{16\pi} \int_\Omega \mathrm{d}^3x \left[\mathrm{e}^{5\phi} \left(16\pi E + \tilde{A}_{ij} \tilde{A}^{ij} - \frac{2}{3} K^2 \right) - \mathrm{e}^{\phi} \tilde{R} \right] \tag{82}$$

$$J_i = \frac{1}{8\pi} \varepsilon_{ij}{}^{k} \int_\Omega \mathrm{d}^3x \left[\mathrm{e}^{6\phi} \left(\tilde{A}^j{}_k + \frac{2}{3} x^j \tilde{D}_k K - \frac{1}{2} x^j \tilde{A}_{\ell n} \partial_k \tilde{\gamma}^{\ell n} + 8\pi x^j S_k \right) \right] \tag{83}$$

on which the user can use the reduction functions provided by `Carpet` to perform the volume integral. We note that `ML_ADMQuantities` inherits directly from the BSSN variables stored in `McLachlan` rather than strictly from the base modules. As the surface terms required when converting a surface integral to a volume integral are neglected, this procedure assumes that the integrals of $\tilde{D}^i \mathrm{e}^{\phi}$ and $\mathrm{e}^{6\phi} \varepsilon_{ij}{}^{k} x^j \tilde{A}^{\ell}{}_k$ over the boundaries of the computational domain vanish. The ADM mass and angular momentum can also be calculated from the variables stored in the base modules using the `Extract` module, as surface integrals [143]

$$M = -\frac{1}{2\pi} \oint \tilde{D}^i \psi \, \mathrm{d}^2 \mathcal{S}_i \tag{84}$$

$$J_i = \frac{1}{16\pi} \varepsilon_{ijk} \oint \left(x^j K^{km} - x^k K^{jm} \right) \mathrm{d}^2 \mathcal{S}_m \tag{85}$$

on a specified spherical surface $\mathcal{S}$, preferably one far from the center of the domain since these quantities are only properly defined when calculated at infinity. The ADM linear momentum is not computed.

There are also the quasi-local measures of mass and angular momentum, from any AHs found during the spacetime. Both `AHFinderDirect` and `AHFinder` output the corresponding irreducible mass M_{ir} derived from the area of the horizon

$$M_{\mathrm{ir}} = \sqrt{\frac{A}{16\pi}}. \tag{86}$$

Note that M_{ir} is smaller than the total mass if the BH is spinning.

The module `QuasiLocalMeasures` implements the calculation of mass and spin multipoles from the isolated and dynamical horizon formalism [176, 177], as well as a number of other proposed formulas for quasi-local mass, linear momentum and angular momentum that have been advanced over the years [178]. Even though there are only a few rigorous proofs that establish the properties of these latter quantities, they have been demonstrated to be surprisingly helpful in numerical simulations (see, e.g., [179]), and are therefore an indispensable tool in numerical relativity. QuasiLocalMeasures takes as input a horizon surface, or any other surface that the user specifies (like a large coordinate sphere) and can calculate useful quantities such as the Weyl or Ricci scalars or the three-volume element of the horizon world tube in addition to physical observables such as mass and momenta. For the calculation of the total mass QuasiLocalMeasures uses the Christodoulou mass [180]

$$M^2 = M_{\rm ir}^2 + \frac{S^2}{4M_{\rm ir}^2}, \tag{87}$$

that takes into account the contributions of the irreducible mass $M_{\rm ir}$ and angular momentum S to the total BH mass M.

Finally, the module HydroAnalysis additionally locates the (coordinate) center of mass as well as the point of maximum rest mass density of a matter field.

5.7.3. Gravitational waves. One of the main goals of numerical relativity to date is modeling gravitational waveforms that may be used in template generation to help analyze data from the various gravitational wave detectors around the globe. The Einstein Toolkit includes modules for extracting gravitational waves via either the Moncrief formalism of a perturbation on a Schwarzschild background or the calculation of the Weyl scalar Ψ_4.

The module Extract uses the Moncrief formalism [181] to extract gauge-invariant wavefunctions $Q^{\times}_{\ell m}$ (see [182]) and $Q^{+}_{\ell m}$ (see [183]) given spherical surfaces of constant coordinate radius. The spatial metric is expressed as a perturbation on Schwarzschild and expanded into a tensor basis of the Regge–Wheeler harmonics [182] described by six standard Regge–Wheeler functions $\left\{c_1^{\times\ell m}, c_2^{\times\ell m}, h_1^{+\ell m}, H_2^{+\ell m}, K^{+\ell m}, G^{+\ell m}\right\}$. From these basis functions the gauge-invariant quantities:

$$Q^{\times}_{\ell m} = \sqrt{\frac{2(\ell+2)!}{(\ell-2)!}}\left[c_1^{\times\ell m} + \frac{1}{2}\left(\partial_r - \frac{2}{r}\right)c_2^{\times\ell m}\right]\frac{S}{r} \tag{88}$$

$$Q^{+}_{\ell m} = \frac{1}{\Lambda}\sqrt{\frac{2(\ell-1)(\ell+2)}{\ell(\ell+1)}}\big(\ell(\ell+1)S(r^2\partial_r G^{+\ell m} - 2h_1^{+\ell m}) \\ + 2rS(H_2^{+\ell m} - r\partial_r K^{+\ell m}) + \Lambda r K^{+\ell m}\big) \tag{89}$$

are calculated, where $S = 1 - 2M/r$ and $\Lambda = (\ell-1)(\ell+2) + 6M/r$. These functions then satisfy the Regge–Wheeler $\left(Q^{\times}_{\ell m}\right)$ and Zerilli $\left(Q^{+}_{\ell m}\right)$ wave equations:

$$\left(\partial_t^2 - \partial_{r^*}^2\right)Q^{\times}_{\ell m} = -S\left[\frac{\ell(\ell+1)}{r^2} - \frac{6M}{r^3}\right]Q^{\times}_{\ell m} \tag{90}$$

$$\left(\partial_t^2 - \partial_{r^*}^2\right)Q^{+}_{\ell m} = -S\Bigg[\frac{1}{\Lambda^2}\left(\frac{72M^3}{r^5} - \frac{12M(\ell-1)(\ell+2)}{r^3}\left(1-\frac{3M}{r}\right)\right) \\ + \frac{\ell(\ell^2-1)(\ell+2)}{r^2\Lambda}\Bigg]Q^{+}_{\ell m}, \tag{91}$$

where $r^* = r + 2M\ln(r/2M - 1)$. Since these functions describe the 4-metric as a perturbation on Schwarzschild, the spacetime must be approximately spherically symmetric for the output to be interpreted as first-order gauge-invariant waveforms.

For more general spacetimes, the module WeylScal4 calculates the complex Weyl scalar $\Psi_4 = C_{\alpha\beta\gamma\delta}\, n^\alpha \bar{m}^\beta n^\gamma \bar{m}^\delta$, which is a projection of the Weyl tensor onto components of a null tetrad. WeylScal4 uses the fiducial tetrad [184], written in 3+1 decomposed form as:

$$\ell^\mu = \tfrac{1}{\sqrt{2}}(u^\mu + \tilde{r}^\mu) \tag{92}$$

$$n^\mu = \frac{1}{\sqrt{2}}(u^\mu - \tilde{r}^\mu) \tag{93}$$

$$m^\mu = \frac{1}{\sqrt{2}}(\tilde{\theta}^\mu + \mathrm{i}\tilde{\phi}^\mu), \tag{94}$$

where u^μ is the unit normal to the hypersurface. The spatial vectors $\{\tilde{r}^\mu, \tilde{\theta}^\mu, \tilde{\phi}^\mu\}$ are created by initializing $\tilde{r}^\mu = \{0, x^i\}$, $\tilde{\phi}^\mu = \{0, -y, x, 0\}$, and $\tilde{\theta}^\mu = \{0, \sqrt{\gamma}\gamma^{ik}\varepsilon_{k\ell m}\phi^\ell r^m\}$, then orthonormalizing starting with $\tilde{\phi}^i$ and invoking a Gram–Schmidt procedure at each step to ensure the continued orthonormality of this spatial triad.

The Weyl scalar Ψ_4 is calculated explicitly in terms of projections of the 4-Riemann tensor onto a null tetrad, such that

$$\begin{aligned}\Psi_4 = {} & \mathcal{R}_{ijk\ell}n^i\bar{m}^jn^k\bar{m}^\ell + 2\mathcal{R}_{0jk\ell}(n^0\bar{m}^jn^k\bar{m}^\ell - \bar{m}^0n^jn^k\bar{m}^\ell) \\ & + \mathcal{R}_{0j0\ell}(n^0\bar{m}^jn^0\bar{m}^\ell + \bar{m}^0n^j\bar{m}^0n^\ell - 2n^0\bar{m}^j\bar{m}^0n^\ell).\end{aligned} \tag{95}$$

For a suitably chosen tetrad, this scalar in the radiation zone is related to the strain of the gravitational waves since

$$h = h_+ - \mathrm{i}h_\times = -\int_{-\infty}^{t}\mathrm{d}t'\int_{-\infty}^{t'}\Psi_4\,\mathrm{d}t''. \tag{96}$$

While the waveforms generated by `Extract` are already decomposed on a convenient basis to separate modes, the complex quantity Ψ_4 is provided by `WeylScal4` as a complex grid function. For this quantity, and any other real or complex grid function, the module `Multipole` interpolates the field $u(t, r, \theta, \phi)$ onto coordinate spheres of given radii and calculates the coefficients

$$C^{\ell m}(t, r) = \int {}_sY^*_{\ell m}u(t, r, \theta, \phi)r^2\,\mathrm{d}\Omega \tag{97}$$

of a projection onto spin-weighted spherical harmonics ${}_sY_{\ell m}$. `Multipole` provides spin-weighted spherical harmonics of arbitrary spin-weight and can thus be used to decompose fields with different spin-weights. For Ψ_4 the appropriate spin-weight is $s = -2$.

5.7.4. Object tracking. We provide a module (`PunctureTracker`) for tracking BH positions evolved with moving puncture techniques. It can be used with (`CarpetTracker`) to have the mesh refinement regions follow the BHs as they move across the grid. The BH position is stored as the centroid of a spherical surface (even though there is no surface) provided by `SphericalSurface`.

Since the punctures only move due to the shift advection terms in the BSSN equations, the puncture location is evolved very simply as

$$\frac{\mathrm{d}x^i}{\mathrm{d}t} = -\beta^i, \tag{98}$$

where x^i is the puncture location and β^i is the shift. Since the puncture location usually does not coincide with grid points, the shift is interpolated to the location of the puncture. Equation (98) is implemented with a simple first-order Euler scheme, accurate enough for controlling the location of the mesh refinement hierarchy.

Another class of objects which often needs to be tracked is NSs. Here is it usually sufficient to locate the position of the maximum density and adapt AMR resolution in these regions accordingly, coupled with the condition that this location can only move at a specifiable maximum speed.

5.7.5. Other analysis modules. The remaining analysis capabilities of the Einstein Toolkit span a variety of primarily vacuum-based functions. First, modules are provided to calculate the Hamiltonian and momentum constraints which are used to monitor how well the evolved spacetime satisfies the Einstein field equations. Two modules, `ADMConstraints` and `ML_ADMConstraints` provide these quantities. Both calculate these directly from variables stored in the base modules described in section 5.2, explicitly written as:

$$H = R - K^i{}_j K^j{}_i + K^2 - 16\pi E \tag{99}$$

$$M_i = \nabla_j K_i{}^j - \nabla_i K - 8\pi S_i, \tag{100}$$

where $S_i = -\frac{1}{\alpha}(T_{i0} - \beta^j T_{ij})$. The difference between these modules lies in how they access the stress–energy tensor $T_{\mu\nu}$, as the module `ADMConstraints` uses a deprecated functionality which does not require storage for $T_{\mu\nu}$.

Finally, `ADMAnalysis` calculates a variety of derived spacetime quantities that are often useful in post-processing such as the determinant of the 3-metric $\det\gamma$, the trace of the extrinsic curvature K, the 3-Ricci tensor in Cartesian coordinates $\mathcal{R}_{ij}$ and its trace $\mathcal{R}$, as well as the 3-metric and extrinsic curvature converted to spherical coordinates.

6. Examples

To demonstrate the properties of the code and its capabilities, we have used it to simulate common astrophysical configurations of interest. Given the community-oriented direction of the project, the parameter files required to launch these simulations and a host of others are included and documented in the code releases[15], along with the data files produced by a representative set of simulation parameters to allow for code validation and confirmation of correct code performance on new platforms and architectures. As part of the internal validation process, nightly builds are checked against a set of benchmarks to ensure that consistent results are generated with the inclusion of all new commits to the code.

The performance of the toolkit for vacuum configurations is demonstrated through evolutions of single, rotating BHs and the merger of binary BH configurations (sections 6.1 and 6.2, respectively). Linear oscillations about equilibrium for an isolated NS are discussed in section 6.3, and the collapse of a NS to a BH, including dynamical formation of a horizon, in section 6.4. Finally, to show a less traditional application of the code, we show its ability to perform cosmological simulations by evolving a Kasner spacetime (see section 6.5).

6.1. Spinning BH

As a first example, we perform simulations of a single distorted rotating BH. We use `TwoPunctures` to set up initial data for a single puncture of mass $M_{\rm bh} = 1$, a spin of dimensionless magnitude $a = S_{\rm bh}/M_{\rm bh}^2 = 0.7$ directed along the z-axis. Evolution of the data is performed by `McLachlan`, AH finding by `AHFinderDirect` and gravitational wave extraction by `WeylScal4` and `Multipole`. Additional analysis of the horizons is done by `QuasiLocalMeasures`. The runs were performed with fixed mesh refinement provided by `Carpet`, using eight levels of refinement on a quadrant grid (symmetries provided by `ReflectionSymmetry` and `RotatingSymmetry180`), centered on the location of the BH. The outer boundaries were placed at $R = 256\,\mathrm{M}$, which places them out of causal contact

[15] The examples shown in this paper are located in a subdirectory `par/arXiv:1111.3344` of the examples obtained when downloading the Einstein Toolkit. The current version of these examples can also be directly accessed via svn checkout https://svn.einsteintoolkit.org/cactus/EinsteinExamples/trunk/par/arXiv:1111.3344.

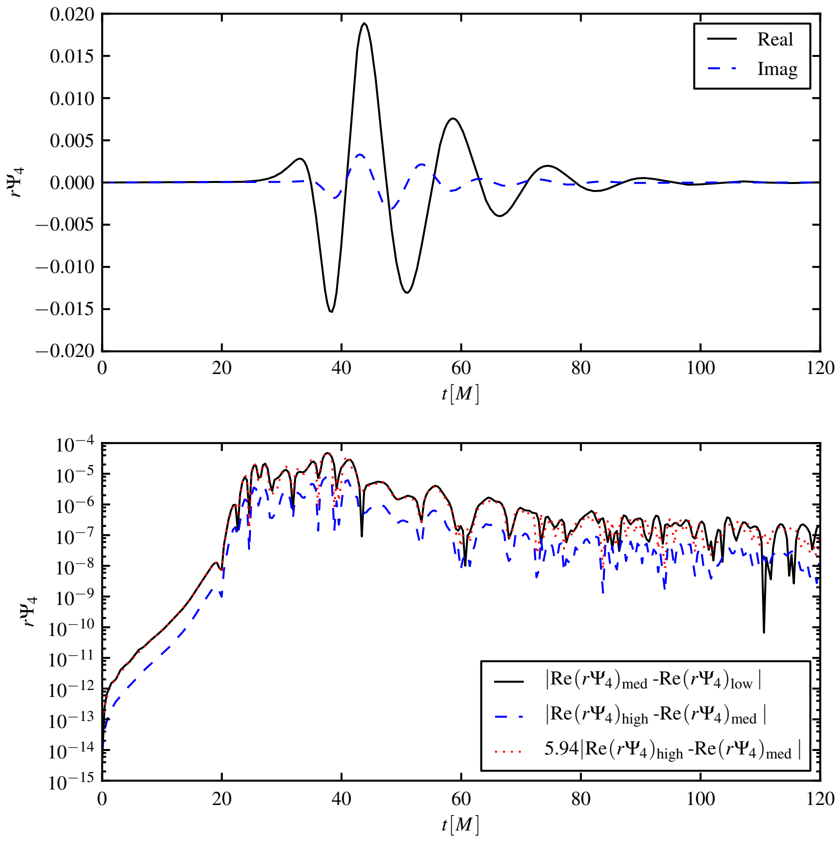


Figure 10. The extracted $\ell = 2$, $m = 0$ mode of Ψ_4 as function of time from the high resolution run (top plot). The extraction was done at $R = 30\,\mathrm{M}$. Shown is both the real (solid black curve) and the imaginary (dashed blue curve) part of the waveform. At the bottom, we show the difference between the medium and low resolution runs (solid black curve), between the high and medium resolution runs (dashed blue curve), and the scaled difference (for fourth-order convergence) between the high and medium resolution runs (dotted red curve) for the real part of the $\ell = 2$, $m = 0$ waveforms.

with the BH during course of the simulation. Since the boundary conditions employed are only approximate, this ensures that the evolution of the BH is not influenced by the imperfect boundary condition. We performed runs at three different resolutions: the low resolution grid spacings were $0.024\,\mathrm{M}/3.072\,\mathrm{M}$, medium were $0.016\,\mathrm{M}/2.048\,\mathrm{M}$ and high were $0.012\,\mathrm{M}/1.536\,\mathrm{M}$, where the numbers refer to the resolution on the finest/coarsest grid. Each refined grid had twice the resolution and half the side length of the containing grid. The runs were performed using the tapering evolution scheme in `Carpet` to avoid interpolation in time during prolongation (i.e. interpolation of data from coarse levels to finer levels). The initial data correspond to a rotating, stationary Kerr BH perturbed by a Brill wave [185] and, as such, have a non-zero gravitational wave content. We evolved the BH using fourth-order finite differencing from $T = 0\,\mathrm{M}$ until it had settled down to a stationary state at $T = 120\,\mathrm{M}$.

Figure 10 shows the $\ell = 2$, $m = 0$ mode of $r\Psi_4$, computed using fourth-order accuracy and extracted at $R = 30\,\mathrm{M}$, and its numerical convergence. In this and in the following sections, a numerical quantity Ψ is said to converge with convergence order Q if for a set of numerical step sizes h_1, h_2, h_3 the difference between successive resolutions scales as

$$\frac{\Psi_{h_1} - \Psi_{h_2}}{\Psi_{h_2} - \Psi_{h_3}} = \frac{h_1^Q - h_2^Q}{h_2^Q - h_3^Q}. \tag{101}$$

In the top plot the black (solid) curve is the real part and the blue (dashed) curve is the imaginary part of $r\Psi_4$ for the high resolution run. Curves from the lower resolution are indistinguishable from the high resolution curve at this scale. In the bottom plot the black (solid) curve shows the absolute value of the difference between the real part of the medium and low resolution waveforms while the blue (dashed) curve shows the absolute value of the difference between the high and medium resolution waveforms in a log-plot. The red (dotted) curve is the same as the blue (dashed) curve, except it is scaled for fourth-order convergence, demonstrating we indeed achieve this. With the resolutions used here this factor is $\left(0.016^4 - 0.024^4\right) / \left(0.012^4 - 0.016^4\right) \approx 5.94$.

Figure 11 shows similar plots for the $\ell = 4, m = 0$ mode of $r\Psi_4$, again extracted at $R = 30\,\mathrm{M}$. The top plot in this case shows only the real part of the extracted waveform but for all three resolutions (black solid curve is high, blue dashed curve is medium and red dotted curve is low resolution). Since the amplitude of this mode is almost a factor of 20 smaller than the $\ell = 2, m = 0$ mode there are actually small differences visible between resolutions in the beginning of the waveform. The bottom plot shows the convergence of the real part of the $\ell = 4, m = 0$ mode (compare with the bottom plot in figure 10) and demonstrates that even though the amplitude is much smaller we still obtain close to perfect fourth-order convergence.

In addition to the modes shown in figures 10 and 11 we note that the extracted $\ell = 4, m = 4$ mode is non-zero due to truncation error, but shows fourth-order convergence to zero with resolution (this mode is not present in the initial data and is not excited during the evolution). Other modes are zero to round-off due to symmetries at all resolutions.

Since there is non-trivial gravitational wave content in the initial data, the mass of the BH changes during its evolution. In figure 12, we show in the top plot the irreducible mass (see equation (86)) as calculated by `AHFinderDirect` as a function of time at the high (black solid curve), medium (blue dashed curve) and low (red dotted curve) resolutions.

Note that the irreducible mass M_{AH} is smaller than the initial mass M_{bh} due to the spin of the BH. The inset shows in more detail the differences between the different resolutions. The irreducible mass increases by about 0.3% during the first 40 M of evolution and then remains constant (within numerical error) for the remainder of the evolution. The bottom plot shows the convergence of the irreducible mass by the difference between the medium and low resolutions (black solid curve), the difference between the high and medium resolutions (blue dashed curved) as well as the scaled difference between the high and medium resolutions for fourth-order (red dotted curve) and third-order (green dash-dotted curve). The convergence is almost perfectly fourth order until $T = 50\,\mathrm{M}$, then better than fourth order until $T = 60\,\mathrm{M}$, and finally between third order and fourth order for the remainder of the evolution. The lack of perfect fourth-order convergence at late times may be attributed to non-convergent errors from the puncture propagating to the horizon location at the lowest resolution.

Finally, in figure 13 we show the total mass M_{IH} (top plot) and the change in the spin, $\Delta S = S(t) - S(t = 0)$, as calculated by `QuasiLocalMeasures`. The total mass M_{IH} is calculated using the Christodoulou formula (see equation (87)) and initially agrees with M_{bh} but then starts to increase due to the flux of gravitational wave energy through the horizon.

In both cases the black (solid) curve is for high, blue (dashed) for medium and red (dotted) for low resolution. Since the spacetime is axisymmetric the gravitational waves cannot radiate angular momentum [186]. Thus any change in the spin must be due to numerical error and

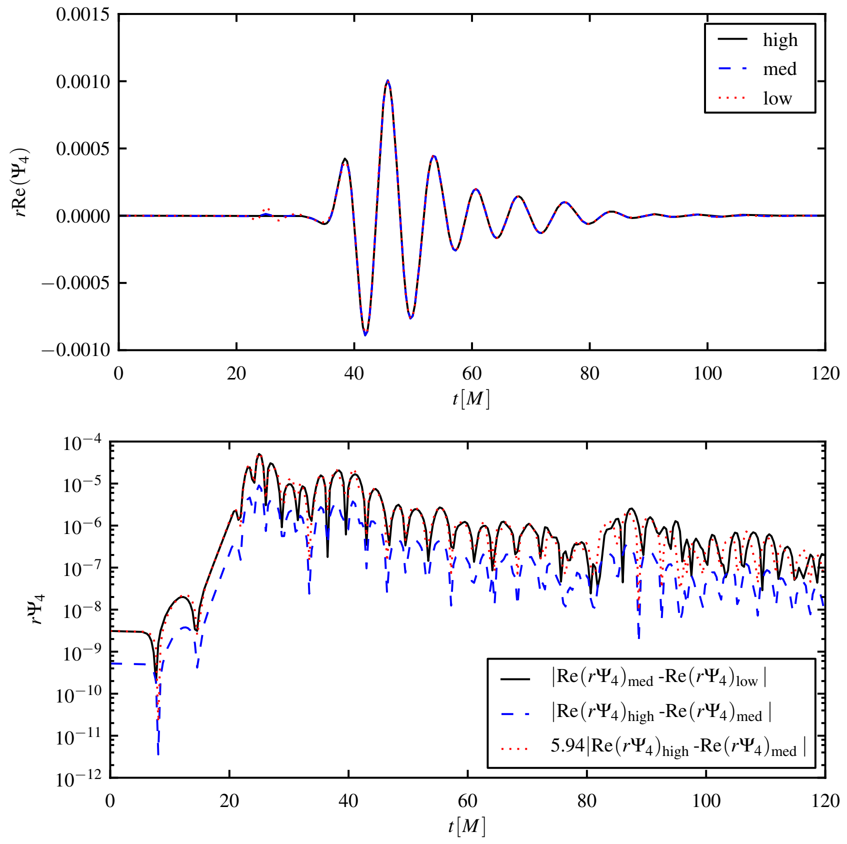
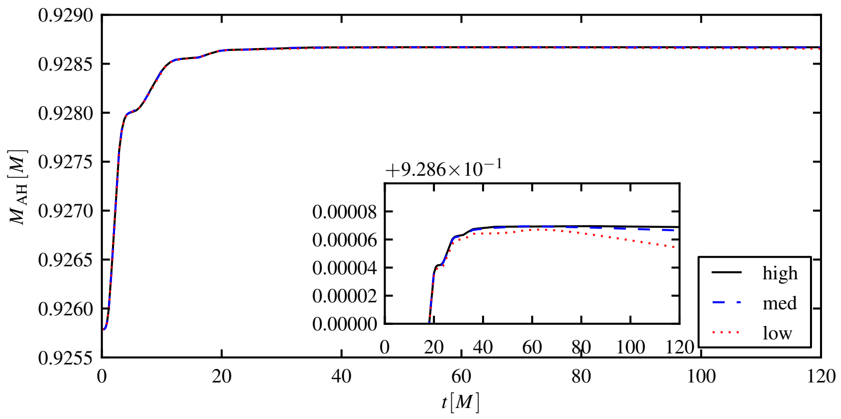


Figure 11. Real part of the extracted $\ell = 4$, $m = 0$ mode of Ψ_4 as function of time (top plot) for the high (solid black curve), medium (dashed blue curve) and low (dotted red curve) resolution runs. The extraction was done at $R = 30\,\mathrm{M}$. The bottom plot shows for the real part of the $\ell = 4$, $m = 0$ waveforms the difference between the medium and low resolution runs (solid black curve), the difference between the high and medium resolution runs (dashed blue curve) as well as the scaled (for fourth-order convergence) difference between the high and medium resolution runs (dotted red curve).

ΔS should converge to zero with increasing resolution. This is clearly shown in the bottom plot of figure 13; the green (dash-dotted) curve (the high resolution result scaled by a factor of 1.78 for second-order convergence to the resolution of the medium resolution) and the blue (dashed) curve are on top of each other. Since the `QuasiLocalMeasures` thorn uses an algorithm which is only second-order accurate overall, this is the expected result. The increase of about 0.22% in the mass of the BH is caused solely by the increase in the irreducible mass.

6.2. BH binary system

To demonstrate the performance in the code for a current problem of wide scientific interest, we have evolved a non-spinning equal-mass BH binary system. The initial data represent a binary system in a quasi-circular orbit, with an initial puncture coordinate separation chosen to be $r = 6\,\mathrm{M}$ so we may track the later inspiral, plunge, merger and ring-down phases of the binary evolution. Table 3 provides more details about the initial binary parameters used to generate the



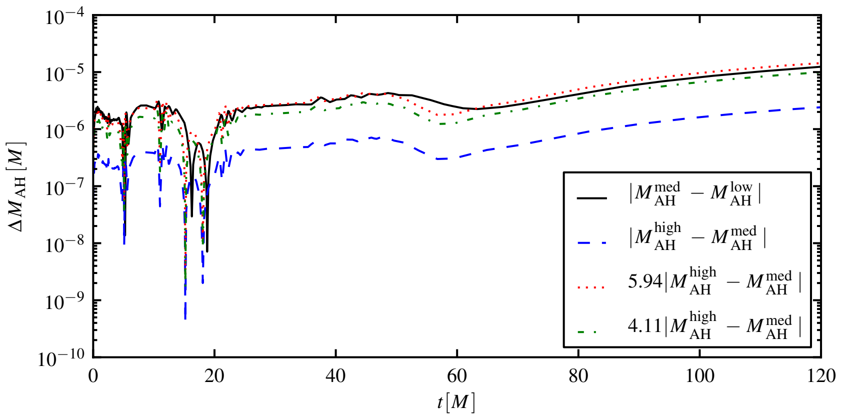


Figure 12. The top plot shows the irreducible mass of the AH as a function of time at low (black solid curve), medium (blue dashed curve) and high (red dotted curve) resolutions. The inset is a zoom in on the y-axis to more clearly show the differences between the resolutions. The bottom plot shows the convergence of the irreducible mass. The black (solid) curve shows the difference between the medium and low resolution results, the blue (dashed) curve shows the difference between the high and medium resolution results. The red (dotted) and green (dash-dotted) show the difference between the high and medium resolutions scaled according to fourth- and third-order convergence, respectively.

Table 3. Initial data parameters and initial ADM mass for a non-spinning equal-mass BH binary system. The punctures are located on the x-axis at positions x_1 and x_2, with puncture bare mass parameters $m_1 = m_2 = m$, and momenta $\vec{p}_1 = -\vec{p}_2 = \vec{p}$.

Configuration	x_1	x_2	p_x	p_y	m	$M_{\rm ADM}$
QC3	3.0	−3.0	0.0	0.138 08	0.476 56	0.984 618

initial data. The `TwoPunctures` module uses these initial parameters to solve (13), the elliptic Hamiltonian constraint for the regular component u of the conformal factor (see section 5.3.3). The spectral solution for this example was determined by using $[n_A, n_B, n_\phi] = [28, 28, 14]$ collocation points, and, along with the Bowen–York analytic solution for the momentum constraints, represents constrained GR initial data $\{\gamma_{ij}, K_{ij}\}$. The evolution is performed by the `McLachlan` module.

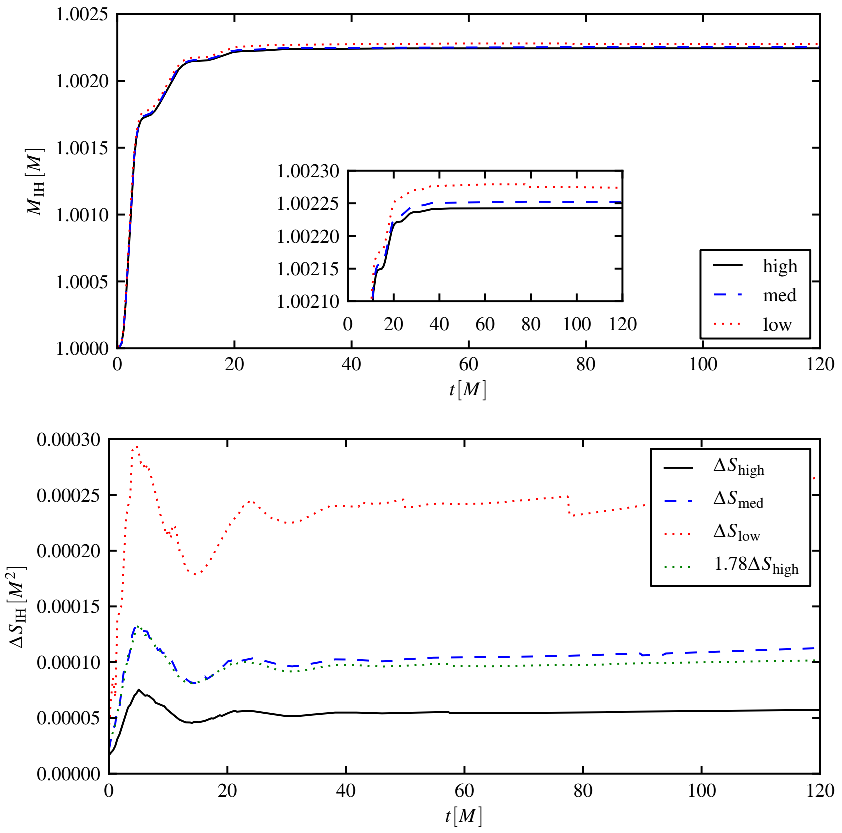


Figure 13. The top plot shows the total mass and the bottom plot shows the change in spin (i.e. $\Delta S = S(t) - S(t = 0)$) of the BH as a function of time. In both plots the black (solid) curve is for high, blue (dashed) for medium and red (dotted) for low resolution. In the bottom plot the green (dash-dotted) curve shows the high resolution result scaled for second-order convergence. The agreement with the medium resolution curve shows that the change in spin converges to zero as expected.

The runs were performed with a moving box mesh refinement provided by `Carpet`, using seven levels of refinement on a quadrant grid (symmetries provided by `ReflectionSymmetry` and `RotatingSymmetry180`). The outer radius is located at $R = 120\,\mathrm{M}$, where M is the sum of the initial ADM masses of the punctures. We performed runs at five different resolutions: 0.03125 M/2.0 M, 0.02344 M/1.5 M, 0.01953 M/1.25 M, 0.01563 M/1.0 M and 0.01172 M/0.75 M, where the numbers refer to the resolution on the finest/coarsest grid. Two sets of moving boxes, each centered on one of the BHs, are used, with the finest grid having a side length of 2 M and each coarser grid having twice the size and half the resolution of the grid it contained.

Figure 14 shows the two puncture tracks throughout all phases of the binary evolution, provided by the `PunctureTracker` module. In the same plot we have recorded the intersection of the AH 2-surface with the $z = 0$ plane every time interval $t = 10\,\mathrm{M}$ during the evolution. A common horizon is first observed at $t = 116\,\mathrm{M}$. These AHs were found by the `AHFinderDirect` module and their radius and location information stored as a 2-surface

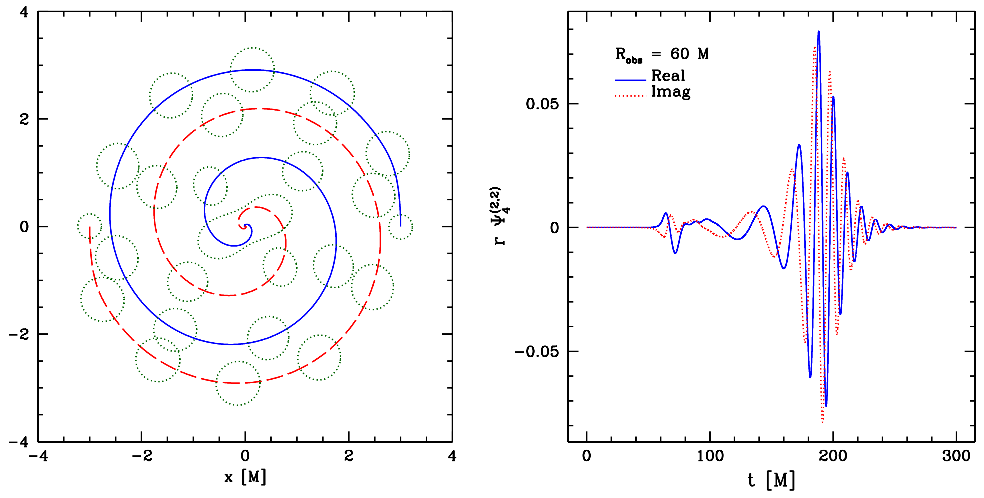


Figure 14. In the left panel, we plot the tracks corresponding to the evolution of two punctures initially located on the x-axis at $x = \pm 3$. The solid blue line represents puncture 1, and the dashed red line puncture 2. The circular dotted green lines are the intersections of the AHs with the $z = 0$ plane plotted every 10 M during the binary evolution. A common horizon appears at $t = 116$ M. In the right panel, we plot the real (solid blue line) and imaginary (dotted red line) parts of the $(l = 2, m = 2)$ mode of the Weyl scalar Ψ_4 as extracted at an observer radius of $R_{\rm obs} = 60$ M.

with spherical topology by the `SphericalSurface` module. The irreducible mass and spin angular momentum of the merged BH were calculated by the `QuasiLocalMeasures` module, and were found to be 0.884 M and 0.625 M^2, respectively.

Figure 14 shows the real and imaginary parts of the $(l = 2, m = 2)$ mode for Ψ_4 extracted on a sphere centered at the origin at $R_{\rm obs} = 60$ M. The number of grid points on the sphere was set to be $[n_\theta, n_\phi] = [120, 240]$, which yields an angular resolution of 2.6×10^{-2} radians, and an error of the same order, since the surface integrals were calculated by midpoint rule—a first-order accurate method. In order to evaluate the convergence of the numerical solution, we ran five simulations with different resolutions, and focus our analysis on the convergence of the phase $\varphi(t)$ and amplitude $A(t)$ of the (2, 2) mode. To take differences between the numerical values at two different grid resolutions, we use an eighth-order accurate Lagrange operator to interpolate the higher-accuracy finite difference solution into the immediately coarser grid. We have experimented with fourth and sixth order as well, to evaluate the level of noise these interpolations could potentially introduce, but did not observe any noticeable difference and we report here on results from the higher order option.

In figure 15, we show the convergence of the amplitude and phase of the Weyl scalar by plotting the logarithm of the absolute value of the differences between two levels of resolution. The differences clearly converge to zero as the resolution is increased. We also indicate on both plots the time at which the gravitational wave frequency reaches $\omega = 0.2/$M. This reference time has been suggested, for example in the NRAR [187] collaboration, as a useful point at which errors can be compared between codes. The suggestion is that the accumulated phase error should not be larger than 0.05 radians at this reference time, and that this is roughly the scale of accuracy which might be useful for comparisons with analytic approximation methods. From the plot, we see that the phase difference between the higher and high resolutions and the one between high and medium-high resolutions satisfy this criterion, while the phase

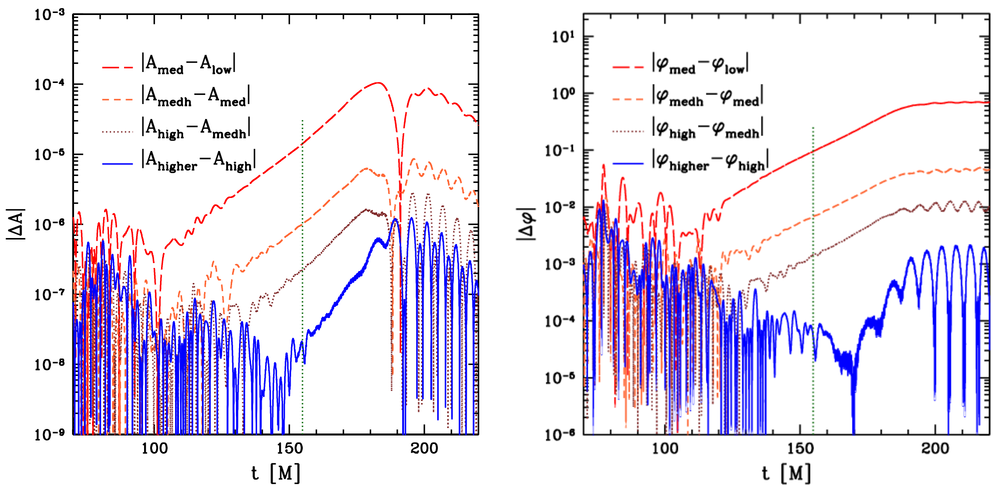


Figure 15. Weyl scalar amplitude (left panel) and phase (right panel) convergence. The long dashed red curves represent the difference between the medium and low-resolution runs. The short dashed orange curves show the difference between the medium-high and medium resolution runs. The dotted brown ones, the difference between high and medium-high resolutions, while the solid blue curves represent the difference between the higher and high resolution runs. The dotted vertical green line at $t = 154\,\mathrm{M}$ indicates the point during the evolution at which the Weyl scalar frequency reaches $\omega = 0.2\mathrm{M}^{-1}$. Observe that the three highest resolutions accumulate a phase error below the standard of 0.05 radians required by the NRAR collaboration.

difference between the medium-high and medium resolutions barely satisfies the criterion; and the one between medium and low resolutions does not. We conclude that the three highest resolution runs do have the suggested accuracy. While the waveform shown here is too short for comparison with analytic methods, we illustrate here that these sorts of comparisons can be made using the toolkit. Note that simulations starting from greater BBH separation leading to longer waveforms will require significantly more resolution to achieve the same phase error.

6.3. Linear oscillations of TOV stars

The examples in the previous subsections did not include the evolution of matter within a relativistic spacetime. One interesting test of a coupled matter-spacetime evolution is to measure the eigenfrequencies of a stable TOV star (see, e.g., [188–192]). These eigenfrequencies can be compared to values known from linear perturbation theory.

We begin our simulations with a self-gravitating fluid sphere, described by a polytropic EOS. This one-dimensional initial-data set is obtained by the code described in section 5.3.4, and is interpolated on the three-dimensional, computational evolution grid. This system is then evolved using the BSSN evolution system implemented in `McLachlan` and the hydrodynamics evolution system implemented in `GRHydro`.

For the convergence test described here, we set up a stable TOV star described by a polytropic EOS (71) with $K = 100$ and $\Gamma = 2$, and an initial central density of $\rho_c = 1.28 \times 10^{-3}$. This model can be taken to represent a non-rotating NS with a mass of $M = 1.4\,\mathrm{M}_{\odot}$.

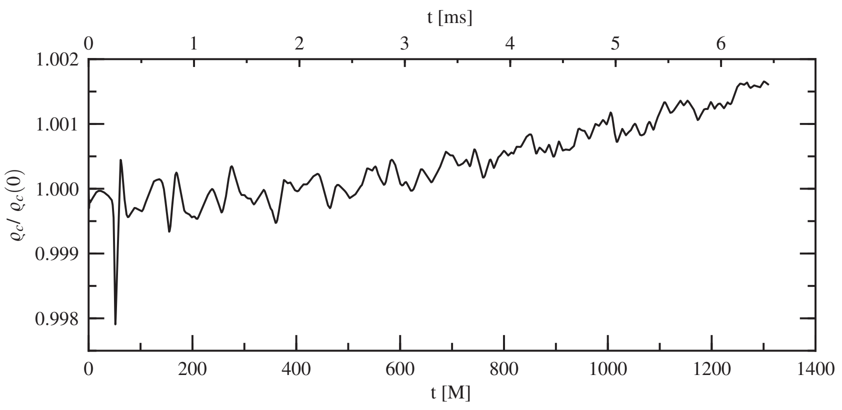


Figure 16. Evolution of the central density for the TOV star. Clearly visible is an initial spike, produced by the mapping of the one-dimensional equilibrium solution onto the three-dimensional evolution grid. The remainder of the evolution however, the central density evolution is dominated by continuous excitations coming from the interaction of the stellar surface with the artificial atmosphere.

The runs are performed with fixed mesh refinement provided by `Carpet`, using 5 levels of refinement on a quadrant grid (symmetries provided by `ReflectionSymmetry` and `RotatingSymmetry180`) The outer boundaries are placed at $R = 640\,\mathrm{M}$, and refined boxes are centered around the star at the origin, each doubling the resolution, with sizes of 240 M, 120 M, 60 M and 30 M. We perform runs at 3 different resolutions: the low resolution is $0.500\,\mathrm{M}/8.0\,\mathrm{M}$, medium was $0.250\,\mathrm{M}/4.0\,\mathrm{M}$ and high was $0.125\,\mathrm{M}/2.0\,\mathrm{M}$, where the numbers refer to the resolution on the finest/coarsest grid. Each refined grid had twice the resolution and half the side length of the containing grid, with the finest grid completely covering the star.

In figure 16 we show the evolution of the central density of the star over an evolution time of 1300 M (6.5 ms). The initial spike is due to the perturbation of the solution resulting from the mapping onto the evolution grid. The remaining oscillations are mainly due to the interaction of the star and the artificial atmosphere and are present during the whole evolution. Given enough evolution time, the frequencies of these oscillations can be measured with satisfactory accuracy.

In figure 17 we show the power spectral density (PSD) of the central density oscillations computed from a full 3D relativistic hydrodynamics simulation, compared to the corresponding frequencies as obtained with perturbative techniques (kindly provided by Kentaro Takami and computed using the method described in [193])[16]. The agreement of the fundamental mode and first three overtone frequencies is clearly visible, but is limited beyond this by the finite numerical resolution. Higher overtones should be measurable with higher resolution, but at substantial computational cost.

Within this test it is also interesting to study the convergence behavior of the coupled curvature and matter evolution code. One of the variables often used for this test is the Hamiltonian constraint violation (99). This violation vanishes for the continuum problem, but is non-zero and resolution-dependent in discrete simulations. The expected rate of convergence of the hydrodynamics code, as defined in (101), lies between 1 and 2. It cannot be higher than

[16] The PSD was computed using the entire time series of the high-resolution run, by removing the linear trend and averaging over Hanning windows overlapping half the signal length after padding the signal to five times its length.

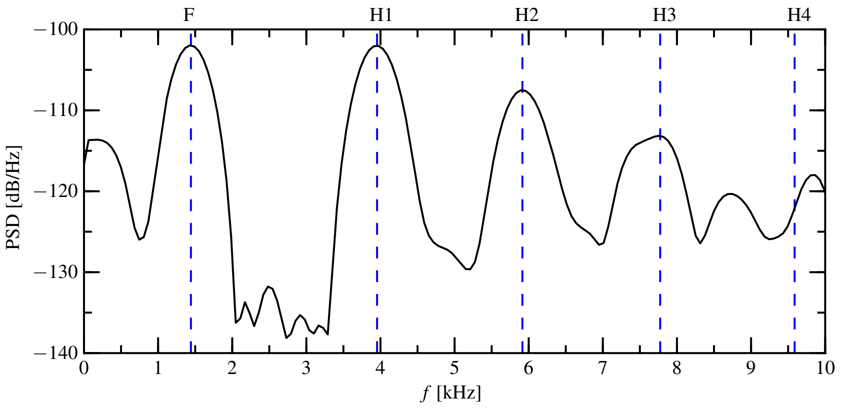


Figure 17. Eigenfrequency mode spectrum of a TOV star. Shown is the PSD of the central matter density, computed from a full 3D relativistic hydrodynamics simulation and compared to the values obtained by perturbation theory. The agreement of the frequencies of the fundamental mode and the first three overtones is clearly visible.

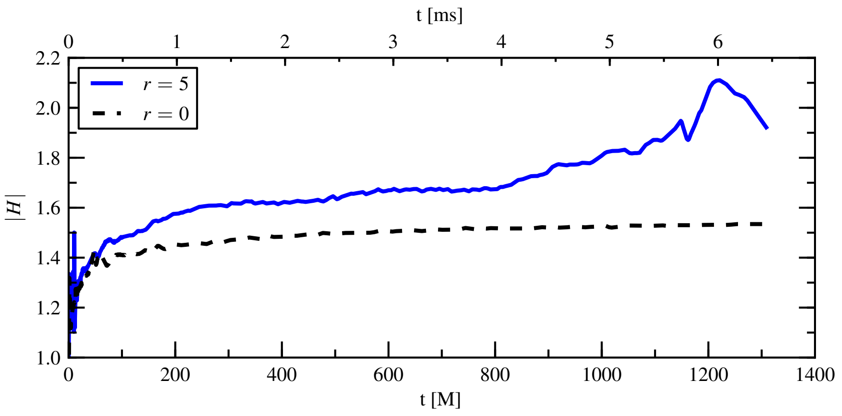


Figure 18. Convergence factor of Hamiltonian constraint violation at $r = 0$ M and $r = 5$ M. The observed convergence order of about 1.5 at the center of the star is lower then the general second order of the hydrodynamics evolution scheme. This is expected because the scheme's convergence rate drops to first order at extrema or shocks, like the stellar center or surface. Consequently, the observed convergence order about half way between the stellar center and surface is higher than 1.5, but mostly below 2.

2 due to the HRSC scheme used which is of second order. Depending on solution itself, the hydrodynamics code is only of first order in particular regions, e.g., at extrema (like the center of the star), or at the stellar surface.

Figure 18 shows the order of convergence of the Hamiltonian constraint violation, using the two highest-resolution runs, at the stellar center and a coordinate radius of $r = 5$ M which is about half way between the center and the surface. The observed convergence rate for most of the simulation time lies between 1.4 and 1.5 at the center, and between 1.6 and 2 at $r = 5$ M, consistent with the expected data-dependent convergence order of the underlying hydrodynamics evolution scheme.

6.4. NS collapse

The previous examples dealt either with preexisting BHs, either single or in a binary, or with a smooth singularity free spacetime, as in the case of the TOV star. The evolution codes in the toolkit are, however, also able to handle the dynamic formation of a BH, that is to follow a NS collapse into a BH. As a simple example of this process, we study the collapse of a non-rotating TOV star. We create initial data as in section 6.3 using $\rho_c = 3.154 \times 10^{-3}$ and $K_{\mathrm{ID}} = 100$, $\Gamma = 2$, yielding a star model of gravitational mass $M = 1.67\,\mathrm{M}_{\odot}$, that is at the onset of instability. As it is common in such situations (e.g., [101]), we trigger collapse by reducing the pressure support after the initial data have been constructed by lowering the polytropic constant K_{ID} from its initial value to $K = 0.98\,K_{\mathrm{ID}} = 98$. To ensure that the pressure-depleted configuration remains a solution of the Einstein constraint equations in the presence of matter, we rescale the rest mass density ρ such that the total energy density E defined in (99) does not change:

$$\rho' + K(\rho')^2 = \rho + K_{\mathrm{ID}}\rho^2. \tag{102}$$

Compared to the initial configuration, this rescaled star possesses a slightly higher central density and lower pressure. This change in K accelerates the onset of collapse that would otherwise rely on being triggered by numerical noise, which would not be guaranteed to converge to a unique solution with increasing resolution.

The runs are performed with fixed mesh refinement provided by `Carpet`, using eight levels of refinement on a quadrant grid (symmetries provided by `ReflectionSymmetry`), centered on the star. Refined regions are centered around the star at the origin with the finest level having a side length of 2 M and each coarser grid (with the exception of the coarsest level) having twice the side length and half the resolution of the contained grid. This places the two finest levels inside of the star; they are used to resolve the high density region during collapse. The outer boundaries were placed at $R = 204.8\,\mathrm{M}$. We perform runs at four different resolutions: from lowest to highest the resolution are 0.025 M/3.2 M, 0.0188 M/2.4 M, 0.0125 M/1.6 M and 4.67×10^{-3} M/0.6 M, where the numbers refer to the resolution on the finest/coarsest grid.

We use the PPM reconstruction method and the HLLE Riemann solver (see sections 5.5.2 and 5.5.3, respectively) to obtain second-order convergent results in smooth regions. Due to the presence of the density maximum at the center of the star and the non-smooth atmosphere at the edge of the star, we expect the observed convergence rate to be somewhat lower than second order, but higher than first order.

In figure 19, we plot the approximate coordinate size of the star as well as the mean coordinate radius of the AH that eventually forms in the simulation. The AH is first found at approximately the time when the star's coordinate radius is twice the size of the forming AH [194]. At this point the majority of the matter is inside the AH, whose size quickly grows to its final size while the numerical spacetime approaches the final 'trumpet' configuration.

In figure 20, we display the convergence factor (101) for the Hamiltonian constraint violation (99) at the center of the collapsing star. Up to the time when the AH forms, the convergence order is an expected ≈ 1.5. At later times the Γ-driver shift condition used pushes the infalling matter beyond the innermost grid point [194], which prevents a clean measurement of a convergence factor. Eventually the shock-capturing scheme is unable to resolve the steep gradients and becomes dissipative, which leads to a loss of rest mass inside of the AH. The observed horizon mass on the other hand stays constant.

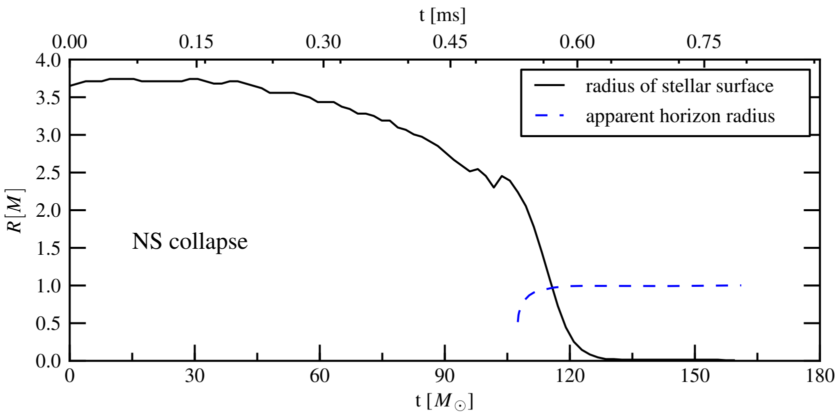


Figure 19. Coordinate radius of the surface of the collapsing star and radius of the forming AH. The stellar surface is defined as the point where ρ is 100 times the atmosphere density. R is the mean coordinate radius of the AH. The lower x-axis displays time in code units where $M_\odot = G = c = 1$, and the upper x-axis shows the corresponding physical time using 1 $M_\odot = 4.93\ \mu$s.

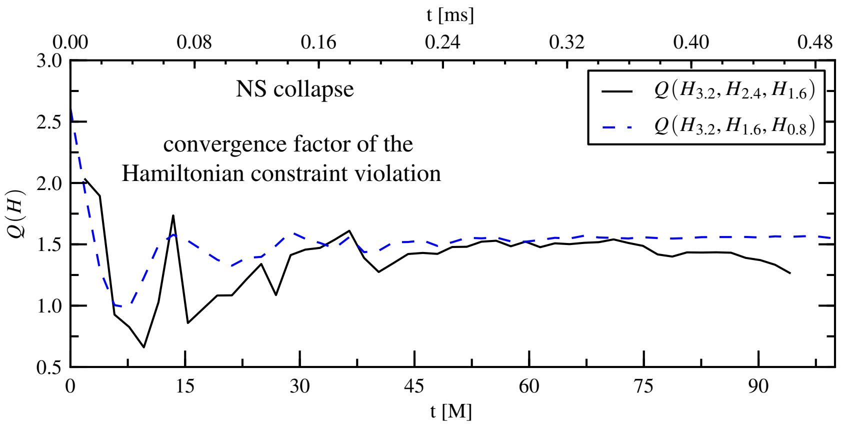


Figure 20. Convergence factor for the Hamiltonian constraint violation at the center of the collapsing star. We plot convergence factors computed using a set of four runs covering the diameter of the star with ≈ 60, 80, 120, and 240 grid points. The units of time on the upper and lower x-axes are identical to those of figure 19.

6.5. Cosmology

The Einstein Toolkit is not only designed to evolve compact-object spacetimes, but also to solve the initial-value problem for spacetimes with radically different topologies and global properties. In this section, we illustrate the evolution of an initial-data set representing a constant-t section of a spacetime from the Gowdy T^3 class [195, 196], namely the Kasner model. This spacetime has the line element:

$$ds^2 = -dt^2 + t^{4/3}(dx^2 + dy^2) + t^{-2/3}\,dz^2 \tag{103}$$

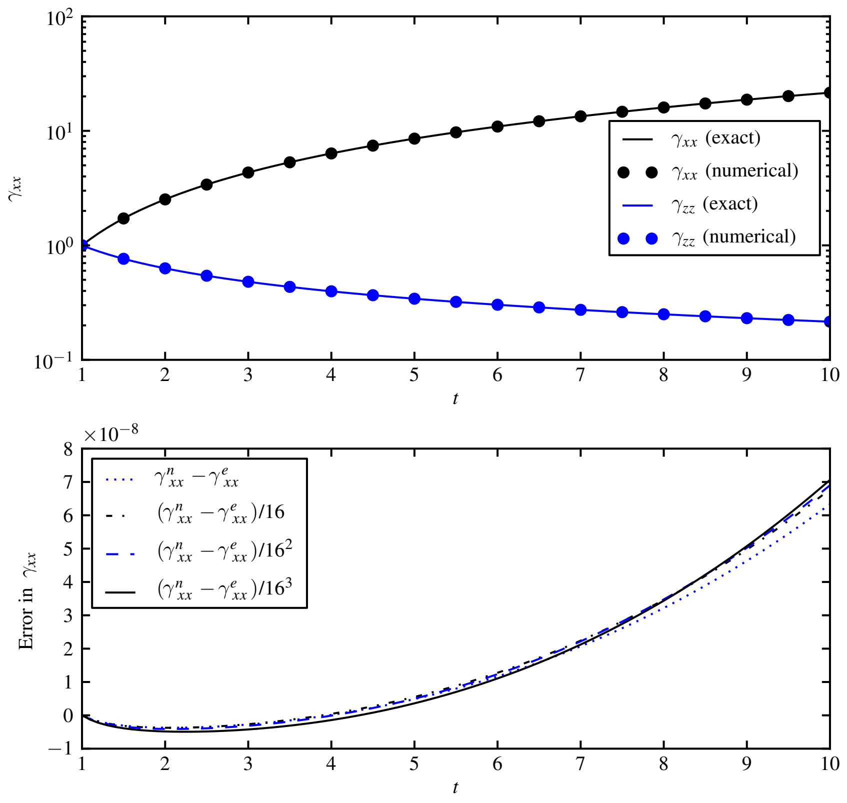


Figure 21. Top: the evolution of a vacuum spacetime of the type (103); the initial data are chosen as $\gamma_{ij} = \delta_{ij}$ and $K_{ij} = \mathrm{diag}(-2/3, -2/3, 1/3)$. Bottom: the numerical error for the sequence of four time resolutions [0.0125, 0.025, 0.05, 0.1]; the superscripts n and e indicate the numerical and the exact solutions, respectively, and the errors are scaled according to the expectation for fourth-order convergence.

defined on a 3-torus $-x_0 \leqslant x \leqslant x_0$, $-y_0 \leqslant y \leqslant y_0$, $-z_0 \leqslant z \leqslant z_0$, with periodic boundary conditions. In the 3+1 decomposition described in section 5.2.1, this reads:

$$\alpha(t) = 1 \tag{104}$$

$$\beta^i(t) = 0 \tag{105}$$

$$\gamma_{ij}(t) = \mathrm{diag}(t^{4/3}, t^{4/3}, t^{-2/3}) \tag{106}$$

$$K_{ij}(t) = \mathrm{diag}\left(-\tfrac{2}{3}\,t^{1/3}, -\tfrac{2}{3}\,t^{1/3}, \tfrac{1}{3}\,t^{-5/3}\right). \tag{107}$$

This solution represents a vacuum, expanding universe with an homogeneous but anisotropic metric tensor. In figure 21, we show the full evolution of the $t = 1$ slice of spacetime (103), along with the associated error for a sequence of time resolutions. We choose $x_0 = y_0 = z_0 = 5$ and spatial resolution equal to 1, and we run a set of four time resolutions equal to [0.0125, 0.025, 0.05, 0.1], finding the expected fourth-order convergence (101).

7. Conclusion and future work

The work presented here is but a snapshot of the Einstein Toolkit's ongoing development, whose ultimate goal is to provide an open-source set of robust baseline codes to realistically and reproducibly model the whole spectrum of relativistic astrophysical phenomena including, but not limited to, isolated BHs and NSs, binary BH coalescence in vacuum and gaseous environments, double NS and NS—BH mergers, core-collapse supernovae, and gamma-ray bursts.

For this, much future work toward including proper treatments of magnetic fields, more complex EOSs, nuclear reactions, neutrinos, and photons will be necessary and will need to be matched by improvements in infrastructure (e.g., more flexible AMR on general grids) and computing hardware for the required fully coupled 3-D, multi-scale, multi-physics simulations to become reality. These tasks, as well as the others mentioned below, are likely to occupy a great deal of the effort spent developing future versions of the Einstein Toolkit over the next few years.

Without a doubt, collapsing stars and merging BH–NS and NS–NS binaries must be simulated with GRMHD to capture the effects of magnetic fields that in many cases will alter the simulation outcome on a qualitative level and may be the driving mechanisms behind much of the observable EM signature from GRBs (e.g., [197]) and magneto-rotationally exploding core-collapse supernovae (e.g., [198]). To date, all simulations that have taken magnetic fields into account are still limited to the ideal MHD approximation, which assumes perfect conductivity. Non-ideal GRMHD schemes are just becoming available (see, e.g., [199, 200]), but have yet to be implemented widely in many branches of numerical relativity.

Most currently published 3D GR(M)HD simulations, with the exception of recent work on massive star collapse (see, e.g., [93]) and binary mergers (see, e.g., [54]), relied on simple zero-temperature descriptions of NS stellar structure, with many assuming simple polytropic forms. Such EOSs are computationally efficient, but are not necessarily a good description for matter in relativistic astrophysical systems. The inclusion of finite-temperature EOSs, derived from the microphysical descriptions of high-density matter, will lead to qualitatively different and much more astrophysically reliable results (see, e.g., [93]). In addition, most GR(M)HD studies neglect transport of neutrinos and photons and their interactions with matter. Neutrinos in particular play a crucial role in core-collapse supernovae and in the cooling of NS–NS merger remnants, thus they must not be left out when attempting to accurately model such events. Few studies have incorporated neutrino and/or photon transport and interactions in approximate ways (see, e.g., [54, 72, 93, 201]).

Besides the addition of new physics modules, existing techniques require improvement. One example is the need for the gauge-invariant extraction of gravitational waves from simulation spacetimes as realized by the cauchy characteristic extraction (CCE) technique recently studied in [118, 202, 203]. The authors of one such CCE code [202] have agreed to make their work available to the whole community by integrating their CCE routines into the Einstein Toolkit release 2011_11 'Maxwell', which will be described elsewhere.

A second much needed improvement of our existing methods is a transition to cell-centered AMR for GR hydrodynamic simulations, which would allow for exact flux conservation across AMR interfaces via a refluxing step that adjusts coarse and/or fine grid fluxes for consistency (e.g., [117]). This is also a prerequisite for the constrained transport method [204] for ensuring the divergence-free condition for the magnetic field in a future implementation of GRMHD within the Einstein Toolkit. Work toward cell-centered AMR, refluxing, and GRMHD is underway and will be reported in a future publication.

While AMR can increase resolution near regions of interest within the computational domain, it does not increase the convergence order of the underlying numerical methods. Simulations of BHs can easily make use of high-order numerical methods, with eighth-order convergence common at present. However, most GRMHD schemes, though they implement HRSC methods, are limited to second-order numerical accuracy in the hydrodynamic/MHD sector while performing curvature evolution with fourth-order accuracy or more. Higher order GRMHD schemes are used in fixed-background simulations (e.g., [205]), but still await implementation in fully dynamical simulations.

Yet another important goal is to increase the scalability of the `Carpet` AMR infrastructure. As we have shown, good scaling is limited to only a few thousand processes for some of the most widely used simulation scenarios. Work is in progress to eliminate this bottleneck [206]. On the other hand, a production simulation is typically composed of a large number of components, and even analysis and I/O routines have to scale well to achieve overall good performance. This is a highly non-trivial problem, since most Einstein Toolkit physics module authors are neither computer scientists nor have they had extensive training in parallel development and profiling techniques. Close collaboration with experts in these topics has been fruitful in the past and will be absolutely necessary for the optimization of Einstein Toolkit codes for execution on the upcoming generation of true petascale supercomputers on which typical compute jobs are expected to be running on 100 000 and more compute cores.

Acknowledgments

The authors wish to thank Ed Seidel whose inspiration and vision has driven work toward the Einstein Toolkit over the past 15 years. The authors are also grateful to the large number of people who contributed to the Einstein Toolkit via ideas, code, documentation, and testing; without these contributions, this toolkit would not exist today. The authors thank their referees for their diligence and their many helpful suggestions. The Einstein Toolkit is directly supported by the National Science Foundation in the USA under the grant nos 0903973/0903782/0904015 (CIGR). Related grants contribute directly and indirectly to the success of CIGR, including NSF OCI-0721915, NSF OCI-0725070, NSF OCI-0832606, NSF OCI-0905046, NSF OCI-0941653, NSF AST-0855535, NSF DMS-0820923, NASA 08-ATFP08-0093, EC-FP7 PIRG05-GA-2009-249290 and Deutsche Forschungsgemeinschaft grant SFB/Transregio 7 'Gravitational-Wave Astronomy'. Results presented in this paper were obtained through computations on the Louisiana Optical Network Initiative under allocation loni_cactus05 and loni_numrel07, as well as on NSF XSEDE under allocations TG-MCA02N014, TG-PHY060027N, TG-PHY100033, at the National Energy Research Scientific Computing Center (NERSC), which is supported by the Office of Science of the US Department of Energy under contract DE-AC03-76SF00098, at the Leibniz Rechenzentrum of the Max Planck Society, and on Compute Canada resources via project cfz-411-aa. GA acknowledges that this material is based upon work supported while serving at the National Science Foundation. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Shibata M and Uryu K 2000 Simulation of merging binary neutron stars in full general relativity: Gamma = two case *Phys. Rev.* D **61** 064001

[2] Shibata M and Uryu K 2002 Gravitational waves from the merger of binary neutron stars in a fully general relativistic simulation *Prog. Theor. Phys.* **107** 265

[3] Shibata M, Taniguchi K and Uryu K 2003 Merger of binary neutron stars of unequal mass in full general relativity *Phys. Rev.* D **68** 084020

[4] Shibata M, Taniguchi K and Uryu K 2005 Merger of binary neutron stars with realistic equations of state in full general relativity *Phys. Rev.* D **71** 084021

[5] Shibata M and Taniguchi K 2006 Merger of binary neutron stars to a black hole: disk mass, short gamma-ray bursts, and quasinormal mode ringing *Phys. Rev.* D **73** 064027
[6] Pretorius F 2005 Evolution of binary black hole spacetimes *Phys. Rev. Lett.* **95** 121101
[7] Campanelli M, Lousto C O, Marronetti P and Zlochower Y 2006 Accurate evolutions of orbiting black-hole binaries without excision *Phys. Rev. Lett.* **96** 111101
[8] Baker J G, Centrella J, Choi D I, Koppitz M and van Meter J 2006 Gravitational wave extraction from an inspiraling configuration of merging black holes *Phys. Rev. Lett.* **96** 111102
[9] Shibata M and Nakamura T 1995 Evolution of three-dimensional gravitational waves: harmonic slicing case *Phys. Rev.* D **52** 5428–44
[10] Baumgarte T W and Shapiro S L 1999 On the numerical integration of Einstein's field equations *Phys. Rev.* D **59** 024007
[11] Gonzalez J A, Sperhake U, Brügmann B, Hannam M D and Husa S 2007 Total recoil: the maximum kick from nonspinning black-hole binary inspiral *Phys. Rev. Lett.* **98** 091101
[12] Baker J G, Centrella J, Choi D I, Koppitz M, van Meter J R and Miller M C 2006 Getting a kick out of numerical relativity *Astrophys. J.* **653** L93–6
[13] Gonzalez J A, Hannam M D, Sperhake U, Brügmann B and Husa S 2007 Supermassive kicks for spinning black holes *Phys. Rev. Lett.* **98** 231101
[14] Campanelli M, Lousto C O, Zlochower Y and Merritt D 2007 Large merger recoils and spin flips from generic black-hole binaries *Astrophys. J.* **659** L5–8
[15] Holley-Bockelmann K, Gultekin K, Shoemaker D and Yunes N 2008 Gravitational wave recoil and the retention of intermediate mass black holes *Astrophys. J.* **686** 829–37
[16] Pollney D *et al* 2007 Recoil velocities from equal-mass binary black-hole mergers: a systematic investigation of spin–orbit aligned configurations *Phys. Rev.* D **76** 124002
[17] Lousto C O and Zlochower Y 2008 Further insight into gravitational recoil *Phys. Rev.* D **77** 044028
[18] Lousto C O and Zlochower Y 2009 Modeling gravitational recoil from precessing highly-spinning unequal-mass black-hole binaries *Phys. Rev.* D **79** 064018
[19] Baker J G, van Meter J R, McWilliams S T, Centrella J and Kelly B J 2007 Consistency of post-Newtonian waveforms with numerical relativity *Phys. Rev. Lett.* **99** 181101
[20] Husa S, Hannam M, Gonzalez J A, Sperhake U and Brügmann B 2008 Reducing eccentricity in black-hole binary evolutions with initial parameters from post-Newtonian inspiral *Phys. Rev.* D **77** 044037
[21] Baumgarte T, Brady P, Creighton J D E, Lehner L, Pretorius F and DeVoe R 2008 Learning about compact binary merger: the interplay between numerical relativity and gravitational-wave astronomy *Phys. Rev.* D **77** 084009
[22] Buonanno A, Cook G B and Pretorius F 2007 Inspiral, merger and ring-down of equal-mass black-hole binaries *Phys. Rev.* D **75** 124018
[23] Hannam M, Husa S, Sperhake U, Brügmann B and Gonzalez J A 2008 Where post-Newtonian and numerical-relativity waveforms meet *Phys. Rev.* D **77** 044020
[24] Boyle M, Barrow D A, Kidder L E, Mroué A H, Pfeiffer H P, Scheel M A, Cook G B and Teukolsky S A 2007 High-accuracy comparison of numerical relativity simulations with post-Newtonian expansions *Phys. Rev.* D **76** 124038
[25] Hannam M, Husa S, Brügmann B and Gopakumar A 2008 Comparison between numerical-relativity and post-Newtonian waveforms from spinning binaries: the orbital hang-up case *Phys. Rev.* D **78** 104007
[26] Gopakumar A, Hannam M, Husa S and Brügmann B 2008 Comparison between numerical relativity and a new class of post-Newtonian gravitational-wave phase evolutions: the non-spinning equal-mass case *Phys. Rev.* D **78** 064026
[27] Campanelli M, Lousto C O, Nakano H and Zlochower Y 2009 Comparison of numerical and post-Newtonian waveforms for generic precessing black-hole binaries *Phys. Rev.* D **79** 084010
[28] Buonanno A, Pan Y, Baker J G, Centrella J, Kelly B J, McWilliams S T and van Meter J R 2007 Towards faithful templates for non-spinning binary black holes using the effective-one-body approach *Phys. Rev.* D **76** 104049
[29] Ajith P *et al* 2008 A template bank for gravitational waveforms from coalescing binary black holes: I. Non-spinning binaries *Phys. Rev.* D **77** 104017
[30] Baker J G, Centrella J, Choi D I, Koppitz M and van Meter J 2006 Binary black hole merger dynamics and waveforms *Phys. Rev.* D **73** 104002
[31] Baker J G, Campanelli M, Pretorius F and Zlochower Y 2007 Comparisons of binary black hole merger waveforms *Class. Quantum Grav.* **24** S25–31
[32] Hannam M *et al* 2009 The Samurai project: verifying the consistency of black- hole-binary waveforms for gravitational-wave detection *Phys. Rev.* D **79** 084025

[33] Campanelli M, Lousto C O and Zlochower Y 2006 Gravitational radiation from spinning-black-hole binaries: the orbital hang up *Phys. Rev.* D **74** 041501
[34] Campanelli M, Lousto C O and Zlochower Y 2006 Spin–orbit interactions in black-hole binaries *Phys. Rev.* D **74** 084023
[35] Campanelli M, Lousto C O, Zlochower Y, Krishnan B and Merritt D 2007 Spin flips and precession in black-hole-binary mergers *Phys. Rev.* D **75** 064030
[36] Herrmann F, Hinder I, Shoemaker D M, Laguna P and Matzner R A 2007 Binary black holes: spin dynamics and gravitational recoil *Phys. Rev.* D **76** 084032
[37] Rezzolla L, Barausse E, Dorband E N, Pollney D, Reisswig C, Seiler J and Husa S 2008 On the final spin from the coalescence of two black holes *Phys. Rev.* D **78** 044002
[38] Berti E, Cardoso V, Gonzalez J A, Sperhake U and Brügmann B 2008 Multipolar analysis of spinning binaries *Class. Quantum Grav.* **25** 114035
[39] Pretorius F and Khurana D 2007 Black hole mergers and unstable circular orbits *Class. Quantum Grav.* **24** S83–108
[40] Sperhake U, Berti E, Cardoso V, González J A, Brügmann B and Ansorg M 2008 Eccentric binary black-hole mergers: the transition from inspiral to plunge in general relativity *Phys. Rev.* D **78** 064069
[41] Hinder I, Vaishnav B, Herrmann F, Shoemaker D and Laguna P 2008 Universality and final spin in eccentric binary black hole inspirals *Phys. Rev.* D **77** 081502
[42] Grigsby J D and Cook G B 2008 Measuring eccentricity in binary black-hole initial data *Phys. Rev.* D **77** 044011
[43] Pfeiffer H P, Brown D A, Kidder L E, Lindblom L, Lovelace G and Scheel M A 2007 Reducing orbital eccentricity in binary black hole simulations *Class. Quantum Grav.* **24** S59–82
[44] Stephens B C, East W E and Pretorius F 2011 Eccentric black hole-neutron star mergers *Astrophys. J.* **737** L5
[45] Centrella J, Baker J G, Kelly B J and van Meter J R 2010 Black-hole binaries, gravitational waves, and numerical relativity *Rev. Mod. Phys.* **82** 3069–119
[46] Font J A 2008 Numerical hydrodynamics and magnetohydrodynamics in general relativity *Living Rev. Rel.* **11** 7 www.livingreviews.org/lrr-2008-7
[47] Hawley J F 2009 MHD simulations of accretion disks and jets: strengths and limitations *Astrophys. Space Sci.* **320** 107–14
[48] Pandharipande V R and Ravenhall D G 1989 Hot nuclear matter *Nuclear Matter and Heavy Ion Collisions* ed M Soyeur, H Flocard, B Tamain and M Porneuf (New York: Plenum) pp 103–32
[49] Douchin F and Haensel P 2001 A unified equation of state of dense matter and neutron star structure *Astron. Astrophys.* **380** 151–67
[50] Akmal A, Pandharipande V and Ravenhall D 1998 The equation of state of nucleon matter and neutron star structure *Phys. Rev.* C **58** 1804–28
[51] Shen H, Toki H, Oyamatsu K and Sumiyoshi K 1998 Relativistic equation of state of nuclear matter for supernova explosion *Prog. Theor. Phys.* **100** 1013
[52] Shen H, Toki H, Oyamatsu K and Sumiyoshi K 1998 Relativistic equation of state of nuclear matter for supernova and neutron star *Nucl. Phys.* A **637** 435–50
[53] Lattimer J M and Swesty F D 1991 A generalized equation of state for hot, dense matter *Nucl. Phys.* A **535** 331–76
[54] Sekiguchi Y, Kiuchi K, Kyutoku K and Shibata M 2011 Gravitational waves and neutrino emission from the merger of binary neutron stars *Phys. Rev. Lett.* **107** 051102
[55] O'Connor E and Ott C D 2010 A new open-source code for spherically-symmetric stellar collapse to neutron stars and black holes *Class. Quantum Grav.* **27** 114103
[56] Anderson M, Hirschmann E W, Lehner L, Liebling S L, Motl P M, Neilsen D, Palenzuela C and Tohline J E 2008 Simulating binary neutron stars: dynamics and gravitational waves *Phys. Rev.* D **77** 024006
[57] Anderson M, Hirschmann E W, Lehner L, Liebling S L, Motl P M, Neilsen D, Palenzuela C and Tohline J E 2008 Magnetized neutron star mergers and gravitational wave signals *Phys. Rev. Lett.* **100** 191101
[58] Baiotti L, Giacomazzo B and Rezzolla L 2008 Accurate evolutions of inspiralling neutron-star binaries: prompt and delayed collapse to black hole *Phys. Rev.* D **78** 084033
[59] Baiotti L, Giacomazzo B and Rezzolla L 2009 Accurate evolutions of inspiralling neutron-star binaries: assessment of the truncation error *Class. Quantum Grav.* **26** 114005
[60] Baiotti L, Damour T, Giacomazzo B, Nagar A and Rezzolla L 2010 Analytic modelling of tidal effects in the relativistic inspiral of binary neutron stars *Phys. Rev. Lett.* **105** 261101
[61] Baiotti L, Damour T, Giacomazzo B, Nagar A and Rezzolla L 2011 Accurate numerical simulations of inspiralling binary neutron stars and their comparison with effective-one-body analytical models *Phys. Rev.* D **84** 024017

[62] Bernuzzi S, Thierfelder M and Bruegmann B 2011 Accuracy of numerical relativity waveforms from binary neutron star mergers and their comparison with post-Newtonian waveforms arXiv:1109.3611 [gr-qc]
[63] Giacomazzo B, Rezzolla L and Baiotti L 2009 Can magnetic fields be detected during the inspiral of binary neutron stars? *Mon. Not. R. Astron. Soc.* **399** L164–8
[64] Giacomazzo B, Rezzolla L and Baiotti L 2011 Accurate evolutions of inspiralling and magnetized neutron-stars: equal-mass binaries *Phys. Rev.* D **83** 044014
[65] Gold R, Bernuzzi S, Thierfelder M, Bruegmann B and Pretorius F 2011 Eccentric binary neutron star mergers arXiv:1109.5128 [gr-qc]
[66] Hotokezaka K, Kyutoku K, Okawa H, Shibata M and Kiuchi K 2011 Binary neutron star mergers: dependence on the nuclear equation of state *Phys. Rev.* D **83** 124008
[67] Kiuchi K, Sekiguchi Y, Shibata M and Taniguchi K 2009 Longterm general relativistic simulation of binary neutron stars collapsing to a black hole *Phys. Rev.* D **80** 064037
[68] Kiuchi K, Sekiguchi Y, Shibata M and Taniguchi K 2010 Exploring binary-neutron-star-merger scenario of short-gamma-ray bursts by gravitational-wave observation *Phys. Rev. Lett.* **104** 141101
[69] Liu Y T, Shapiro S L, Etienne Z B and Taniguchi K 2008 General relativistic simulations of magnetized binary neutron star mergers *Phys. Rev.* D **78** 024012
[70] Rezzolla L, Baiotti L, Giacomazzo B, Link D and Font J A 2010 Accurate evolutions of unequal-mass neutron-star binaries: properties of the torus and short GRB engines *Class. Quantum Grav.* **27** 114105
[71] Rezzolla L, Giacomazzo B, Baiotti L, Granot J, Kouveliotou C and Aloy M A 2011 The missing link: merging neutron stars naturally produce jet-like structures and can power short gamma-ray bursts *Astrophys. J.* **732** L6
[72] Sekiguchi Y, Kiuchi K, Kyutoku K and Shibata M 2011 Effects of hyperons in binary neutron star mergers *Phys. Rev. Lett.* **107** 211101 (arXiv:1110.4442 [astro-ph.HE])
[73] Thierfelder M, Bernuzzi S and Bruegmann B 2011 Numerical relativity simulations of binary neutron stars *Phys. Rev.* D **84** 044012
[74] Yamamoto T, Shibata M and Taniguchi K 2008 Simulating coalescing compact binaries by a new code SACRA *Phys. Rev.* D **78** 064054
[75] Chawla S, Anderson M, Besselman M, Lehner L, Liebling S L, Motl P M and Neilsen D 2010 Mergers of magnetized neutron stars with spinning black holes: disruption, accretion and fallback *Phys. Rev. Lett.* **105** 111101
[76] Duez M D, Foucart F, Kidder L E, Pfeiffer H P, Scheel M A and Teukolsky S A 2008 Evolving black hole-neutron star binaries in general relativity using pseudospectral and finite difference methods *Phys. Rev.* D **78** 104015
[77] Duez M D, Foucart F, Kidder L E, Ott C D and Teukolsky S A 2010 Equation of state effects in black hole-neutron star mergers *Class. Quantum Grav.* **27** 114106
[78] Etienne Z B, Faber J A, Liu Y T, Shapiro S L, Taniguchi K and Baumgarte T W 2008 Fully general relativistic simulations of black hole-neutron star mergers *Phys. Rev.* D **77** 084002
[79] Etienne Z B, Liu Y T, Shapiro S L and Baumgarte T W 2009 General relativistic simulations of black-hole-neutron-star mergers: effects of black-hole spin *Phys. Rev.* D **79** 044024
[80] Foucart F, Duez M D, Kidder L E and Teukolsky S A 2011 Black hole-neutron star mergers: effects of the orientation of the black hole spin *Phys. Rev.* D **83** 024005
[81] Foucart F *et al* 2011 Black hole-neutron star mergers for 10 solar mass black holes *Phys. Rev.* D **85** 044015 (arXiv:1111.1677 [gr-qc])
[82] Kyutoku K, Shibata M and Taniguchi K 2010 Gravitational waves from nonspinning black hole-neutron star binaries: dependence on equations of state *Phys. Rev.* D **82** 044049
[83] Kyutoku K, Okawa H, Shibata M and Taniguchi K 2011 Gravitational waves from spinning black hole-neutron star binaries: dependence on black hole spins and on neutron star equations of state *Phys. Rev.* D **84** 064018
[84] Lackey B D, Kyutoku K, Shibata M, Brady P R and Friedman J L 2011 Extracting equation of state parameters from black hole-neutron star mergers: I. Nonspinning black holes arXiv:1109.3402 [astro-ph.HE]
[85] Löffler F, Rezzolla L and Ansorg M 2006 Numerical evolutions of a black hole-neutron star system in full general relativity: head-on collision *Phys. Rev.* D **74** 104018
[86] Shibata M and Uryu K 2007 Merger of black hole-neutron star binaries in full general relativity *Class. Quantum Grav.* **24** S125–38
[87] Shibata M and Uryu K 2006 Merger of black hole-neutron star binaries: nonspinning black hole case *Phys. Rev.* D **74** 121503
[88] Shibata M and Taniguchi K 2008 Merger of black hole and neutron star in general relativity: tidal disruption, torus mass, and gravitational waves *Phys. Rev.* D **77** 084015

[89] Shibata M, Kyutoku K, Yamamoto T and Taniguchi K 2009 Gravitational waves from black hole-neutron star binaries I: classification of waveforms *Phys. Rev.* D **79** 044030

[90] Shibata M and Kyutoku K 2010 Constraining nuclear-matter equations of state by gravitational waves from black hole–neutron star binaries *Prog. Theor. Phys. Suppl.* **186** 17–25

[91] Faber J 2009 Status of neutron star-black hole and binary neutron star simulations *Class. Quantum Grav.* **26** 114004

[92] Duez M D 2010 Numerical relativity confronts compact neutron star binaries: a review and status report *Class. Quantum Grav.* **27** 114002

[93] Ott C D, Dimmelmeier H, Marek A, Janka H T, Hawke I, Zink B and Schnetter E 2007 3D collapse of rotating stellar iron cores in general relativity with microphysics *Phys. Rev. Lett.* **98** 261101

[94] Ott C D, Dimmelmeier H, Marek A, Janka H T, Zink B, Hawke I and Schnetter E 2007 Rotating collapse of stellar iron cores in general relativity *Class. Quantum Grav.* **24** S139–54

[95] Shibata M and Sekiguchi Y i 2005 Three-dimensional simulations of stellar core collapse in full general relativity: nonaxisymmetric dynamical instabilities *Phys. Rev.* D **71** 024014

[96] Shibata M, Liu Y T, Shapiro S L and Stephens B C 2006 Magnetorotational collapse of massive stellar cores to neutron stars: simulations in full general relativity *Phys. Rev.* D **74** 104026

[97] Shibata M, Baumgarte T W and Shapiro S L 2000 Stability and collapse of rapidly rotating, supramassive neutron stars: 3D simulations in general relativity *Phys. Rev.* D **61** 044012

[98] Duez M D, Liu Y T, Shapiro S L and Stephens B C 2005 Relativistic magnetohydrodynamics in dynamical spacetimes: numerical methods and tests *Phys. Rev.* D **72** 024028

[99] Duez M D, Liu Y T, Shapiro S L, Shibata M and Stephens B C 2006 Collapse of magnetized hypermassive neutron stars in general relativity *Phys. Rev. Lett.* **96** 031101

[100] Baiotti L, Hawke I, Montero P J, Löffler F, Rezzolla L, Stergioulas N, Font J A and Seidel E 2005 Three-dimensional relativistic simulations of rotating neutron star collapse to a Kerr black hole *Phys. Rev.* D **71** 024035

[101] Baiotti L, Hawke I, Rezzolla L and Schnetter E 2005 Gravitational-wave emission from rotating gravitational collapse in three dimensions *Phys. Rev. Lett.* **94** 131101

[102] Baiotti L, De Pietri R, Manca G M and Rezzolla L 2007 Accurate simulations of the dynamical barmode instability in full general relativity *Phys. Rev.* D **75** 044023

[103] Manca G M, Baiotti L, De Pietri R and Rezzolla L 2007 Dynamical non-axisymmetric instabilities in rotating relativistic stars *Class. Quantum Grav.* **24** S171–86

[104] Cactus Computational Toolkit http://www.cactuscode.org/

[105] Djikstra F and van der Steen A J 2006 Integration of two ocean models within Cactus *Concurrency Comput.: Pract. Exp.* **18** 193–202

[106] Camarda K, He Y and Bishop K 2001 A parallel chemical reaction simulation using cactus *Linux Clusters: The HPC Revolution* www.linuxclustersinstitute.org/conferences/archive/2001/PDF/Camarda_KU.pdf

[107] Einstein Toolkit: open software for relativistic astrophysics http://einsteintoolkit.org/

[108] Goodale T, Allen G, Lanfermann G, Massó J, Radke T, Seidel E and Shalf J 2003 The Cactus framework and toolkit: design and applications *5th Int. Conf. Vector and Parallel Processing (Lecture Notes in Computer Science)* (Berlin: Springer) http://edoc.mpg.de/3341

[109] Allen G, Goodale T, Lanfermann G, Radke T, Rideout D and Thornburg J 2011 *Cactus Users Guide* http://www.cactuscode.org/Guides/Stable/UsersGuide/UsersGuide Stable.pdf

[110] Anninos P *et al* 1995 Three-dimensional numerical relativity: the evolution of black holes *Phys. Rev.* D **52** 2059–82

[111] Anninos P, Masso J, Seidel E, Suen W M and Tobias M 1997 Dynamics of gravitational waves in 3D: formulations, methods, and tests *Phys. Rev.* D **56** 842–58

[112] Seidel E and Suen W M 1999 Numerical relativity as a tool for computational astrophysics *J. Comput. Appl. Math.* **109** 493–525

[113] Cactus runs on 131,072 cores on Blue Gene/P at ANL http://cactuscode.org/media/news/BGP-131072/

[114] Schnetter E, Hawley S H and Hawke I 2004 Evolutions in 3-D numerical relativity using fixed mesh refinement *Class. Quantum Grav.* **21** 1465–88

[115] Schnetter E, Diener P, Dorband E N and Tiglio M 2006 A multi-block infrastructure for three-dimensional time-dependent numerical relativity *Class. Quantum Grav.* **23** S553–78

[116] Carpet: adaptive mesh refinement for the Cactus framework http://www.carpetcode.org/

[117] Berger M J and Oliger J 1984 Adaptive mesh refinement for hyperbolic partial differential equations *J. Comput. Phys.* **53** 484 www/livingreviews.org/lrr-2008-7

[118] Reisswig C, Ott C D, Sperhake U and Schnetter E 2011 Gravitational wave extraction in simulations of rotating stellar core collapse *Phys. Rev.* D **83** 064008

[119] Lousto C O and Zlochower Y 2011 Orbital evolution of extreme-mass-ratio black-hole binaries with numerical relativity *Phys. Rev. Lett.* **106** 041101
[120] Thomas M W and Schnetter E 2010 Simulation factory: taming application configuration and workflow on high-end resources *11th IEEE/ACM International Conference on Grid Computing* pp 369–78 (arXiv:1008.4571 [cs.DC])
[121] SimFactory: herding numerical simulations http://simfactory.org/
[122] Husa S, Hinder I and Lechner C 2006 Kranc: a Mathematica application to generate numerical codes for tensorial evolution equations *Comput. Phys. Commun.* **174** 983–1004
[123] Lechner C, Alic D and Husa S 2004 From tensor equations to numerical code—computer algebra tools for numerical relativity *An. Univ. Vest Tim.* (Seria Matematica-Informatica) **42** 3 (arXiv:cs/0411063)
[124] Kranc: Kranc assembles numerical code http://kranccode.org/
[125] Pollney D, Reisswig C, Schnetter E, Dorband N and Diener P 2011 High accuracy binary black hole simulations with an extended wave zone *Phys. Rev.* D **83** 044045
[126] Alcubierre M, Brandt S, Brügmann B, Holz D, Seidel E, Takahashi R and Thornburg J 2001 Symmetry without symmetry: numerical simulation of axisymmetric systems using Cartesian grids *Int. J. Mod. Phys.* D **10** 273–90
[127] Alcubierre M 2008 *Introduction to 3+1 Numerical Relativity* (Oxford: Oxford University Press)
[128] Baumgarte T W and Shapiro S L 2010 *Numerical Relativity: Solving Einstein's Equations on the Computer* (Cambridge: Cambridge University Press)
[129] Arnowitt R L, Deser S and Misner C W 2008 The dynamics of general relativity *Gen. Rel. Grav.* **40** 1997–2027
[130] York J W Jr 1979 Kinematics and dynamics of general relativity *Sources of Gravitational Radiation* ed L L Smarr pp 83–126
[131] Martí J M, Ibáñez J M and Miralles J M 1991 Numerical relativistic hydrodynamics: local characteristic approach *Phys. Rev.* D **43** 3794–801
[132] Banyuls F, Font J A, Ibanez J M, Marti J M and Miralles J A 1997 Numerical 3+1 general relativistic hydrodynamics: a local characteristic approach *Astrophys. J.* **476** 221
[133] Ibáñez J, Aloy M, Font J, Martí J, Miralles J and Pons J 2001 Riemann solvers in general relativistic hydrodynamics *Godunov Methods: Theory and Applications* ed E Toro (New York: Kluwer/Plenum) p 485 (arXiv:astro-ph/9911034v1)
[134] Ansorg M, Brügmann B and Tichy W 2004 A single-domain spectral method for black hole puncture data *Phys. Rev.* D **70** 064011
[135] GSL: the GNU scientific library http://www.gnu.org/software/gsl/
[136] Galassi M, Davies J, Theiler J, Gough B, Jungman G, Alken P, Booth M and Rossi F 2009 *GNU Scientific Library Reference Manual* 3rd edn (Network Theory Ltd)
[137] LORENE: langage objet pour la RElativité NumériquE http://www.lorene.obspm.fr/
[138] Gourgoulhon E, Grandclement P, Taniguchi K, Marck J A and Bonazzola S 2001 Quasiequilibrium sequences of synchronized and irrotational binary neutron stars in general relativity: I. Method and tests *Phys. Rev.* D **63** 064029
[139] York J W Jr 1999 Conformal 'thin sandwich' data for the initial-value problem of General Relativity *Phys. Rev. Lett.* **82** 1350–3
[140] Brill D R 1959 On the positive definite mass of the Bondi–Weber–Wheeler time-symmetric gravitational waves *Ann. Phys.* **7** 466–83
[141] Martí J M and Müller E 2003 Numerical hydrodynamics in special relativity *Living Rev. Rel.* **2** 3 www.livingreviews.org/lrr-2003-7
[142] Brandt S and Brügmann B 1997 Black hole punctures as initial data for general relativity *Phys. Rev. Lett.* **78** 3606–9
[143] Bowen J M and York J W Jr 1980 Time asymmetric initial data for black holes and black hole collisions *Phys. Rev.* D **21** 2047–56
[144] Tolman R C 1939 Static solutions of Einstein's field equations for spheres of fluid *Phys. Rev.* **55** 364–73
[145] Oppenheimer J R and Volkoff G M 1939 On massive neutron cores *Phys. Rev.* **55** 374–81
[146] Grandclement P, Gourgoulhon E and Bonazzola S 2002 Binary black holes in circular orbits: II. Numerical methods and first results *Phys. Rev.* D **65** 044021
[147] Bocquet M, Bonazzola S, Gourgoulhon E and Novak J 1995 Rotating neutron star models with magnetic field *Astron. Astrophys.* **301** 757
[148] Brown J D, Diener P, Sarbach O, Schnetter E and Tiglio M 2009 Turduckening black holes: an analytical and computational study *Phys. Rev.* D **79** 044023

[149] Alcubierre M, Brügmann B, Dramlitsch T, Font J A, Papadopoulos P, Seidel E, Stergioulas N and Takahashi R 2000 Towards a stable numerical evolution of strongly gravitating systems in general relativity: the conformal treatments *Phys. Rev.* D **62** 044034

[150] Marronetti P, Tichy W, Brügmann B, González J and Sperhake U 2008 High-spin binary black hole mergers *Phys. Rev.* D **77** 064010

[151] Alcubierre M, Brügmann B, Diener P, Koppitz M, Pollney D, Seidel E and Takahashi R 2003 Gauge conditions for long term numerical black hole evolutions without excision *Phys. Rev.* D **67** 084023

[152] Müller D and Brügmann B 2010 Toward a dynamical shift condition for unequal mass black hole binary simulations *Class. Quantum Grav.* **27** 114008

[153] Schnetter E 2010 Time step size limitation introduced by the BSSN gamma driver *Class. Quantum Grav.* **27** 167001

[154] Hawke I, Löffler F and Nerozzi A 2005 Excision methods for high resolution shock capturing schemes applied to general relativistic hydrodynamics *Phys. Rev.* D **71** 104006

[155] Baiotti L, Hawke I, Montero P J and Rezzolla L 2003 A new three-dimensional general-relativistic hydrodynamics code *Mem. Soc. Astron. Ital.* **74** S210 (arXiv:1004.3849v1)

[156] Whisky—relativistic hydrodynamics and magnetohydrodynamics http://www.whiskycode.org/

[157] Hyman J M 1976 The method of lines solution of partial differential equations *ERDA Mathematics and Computing Laboratory, Courant Institute of Mathematical Sciences, New York University Technical Report COO-3077-139*

[158] Roe P L 1986 Characteristic-based schemes for the Euler equations *Ann. Rev. Fluid Mech.* **18** 337–65

[159] van Leer B 1977 Towards the ultimate conservative difference scheme. III—upstream-centered finite-difference schemes for ideal compressible flow. IV—a new approach to numerical convection *J. Comput. Phys.* **23** 263–99

[160] Colella P and Woodward P R 1984 The piecewise parabolic method (PPM) for gas dynamical simulations *J. Comput. Phys.* **54** 174–201

[161] Harten A, Engquist B, Osher S and Chakravarthy S R 1987 Uniformly high order accurate essentially non-oscillatory schemes III *J. Comput. Phys.* **71** 231–303

[162] Shu C W 1999 *High Order Methods for Computational Physics* ed T J Barth and H A Deconinck (New York: Springer) pp 439–582

[163] Harten A, Lax P D and van Leer B 1983 On upstream differencing and Godunov-type schemes for hyperbolic conservation laws *SIAM Rev.* **25** 35

[164] Einfeldt B 1988 On Godunov-type methods for gas dynamics *SIAM J. Numer. Anal.* **25** 294–318

[165] Roe P L 1981 Approximate Riemann solvers, parameter vectors, and difference schemes *J. Comput. Phys.* **43** 357–72

[166] Donat R and Marquina A 1996 Capturing shock reflections: an improved flux formula *J. Comput. Phys.* **125** 42–58

[167] Aloy M, Ibanez J, Marti J and Muller E 1999 Genesis: a high resolution code for 3-D relativistic hydrodynamics *Astrophys. J. Suppl.* **122** 151–66

[168] Janka H T, Zwerger T and Moenchmeyer R 1993 Does artificial viscosity destroy prompt type-II supernova explosions? *Astron. Astrophys.* **268** 360–8

[169] Dimmelmeier H, Ott C D, Janka H T, Marek A and Mueller E 2007 Generic gravitational wave signals from the collapse of rotating stellar cores *Phys. Rev. Lett.* **98** 251101

[170] Diener P 2003 A new general purpose event horizon finder for 3D numerical spacetimes *Class. Quantum Grav.* **20** 4901–18

[171] Anninos P, Camarda K, Libson J, Massó J, Seidel E and Suen W M 1998 Finding apparent horizons in dynamic 3-d numerical space-times *Phys. Rev.* D **58** 024003

[172] Gundlach C 1998 Pseudospectral apparent horizon finders: an efficient new algorithm *Phys. Rev.* D **57** 863–75

[173] Thornburg J 2004 A fast apparent-horizon finder for 3-dimensional Cartesian grids in numerical relativity *Class. Quantum Grav.* **21** 743–66

[174] McLachlan, a public BSSN code http://www.cct.lsu.edu/ eschnett/McLachlan/

[175] Yo H J, Baumgarte T W and Shapiro S L 2002 Improved numerical stability of stationary black hole evolution calculations *Phys. Rev.* D **66** 084026

[176] Dreyer O, Krishnan B, Shoemaker D and Schnetter E 2003 Introduction to isolated horizons in numerical relativity *Phys. Rev.* D **67** 024018

[177] Schnetter E, Krishnan B and Beyer F 2006 Introduction to dynamical horizons in numerical relativity *Phys. Rev.* D **74** 024028

[178] Szabados L B 2004 Quasi-local energy-momentum and angular momentum in GR: a review article *Living Rev. Rel.* **7** 4 www.livingreviews.org/lrr-2004-4

[179] Lovelace G, Chen Y, Cohen M, Kaplan J D, Keppel D, Matthews K D, Nichols D A, Scheel M A and Sperhake U 2010 Momentum flow in black-hole binaries: II. Numerical simulations of equal-mass, head-on mergers with antiparallel spins *Phys. Rev.* D **82** 064031
[180] Christodoulou D 1970 Reversible and irreversible transformations in black-hole physics *Phys. Rev. Lett.* **25** 1596–7
[181] Moncrief V 1974 Gravitational perturbations of spherically symmetric systems: I. The exterior problem. *Ann. Phys.* **88** 323–42
[182] Regge T and Wheeler J A 1957 Stability of a Schwarzschild singularity *Phys. Rev.* **108** 1063–9
[183] Zerilli F J 1970 Effective potential for even parity Regge–Wheeler gravitational perturbation equations *Phys. Rev. Lett.* **24** 737–8
[184] Baker J G, Campanelli M and Lousto C O 2002 The Lazarus project: a pragmatic approach to binary black hole evolutions *Phys. Rev.* D **65** 044001
[185] Brandt S R and Seidel E 1996 The evolution of distorted rotating black holes. 3: initial data *Phys. Rev.* D **54** 1403–16
[186] Brandt S R 1996 Rotating black hole spacetimes *PhD Thesis* University of Illinois at Urbana-Champaign Urbana, Illinois http://hdl.handle.net/2142/18837
[187] NRAR: Numerical-relativity and analytical-relativity collaboration https://www.ninja-project.org/doku.php?id=nrar:home
[188] Gourgoulhon E 1991 Simple equations for general relativistic hydrodynamics in spherical symmetry applied to neutron star collapse *Astron. Astrophys.* **252** 651–63
[189] Romero J V, Ibanez J M A, Marti J M A and Miralles J A 1996 A new spherically symmetric general relativistic hydrodynamical code *Astrophys. J.* **462** 839–54
[190] Shibata M, Baumgarte T W and Shapiro S L 1998 Stability of coalescing binary stars against gravitational collapse: hydrodynamical simulations *Phys. Rev.* D **58** 023002
[191] Font J A, Goodale T, Iyer S, Miller M A, Rezzolla L, Seidel E, Stergioulas N, Suen W M and Tobias M 2002 Three-dimensional general relativistic hydrodynamics. 2. Long term dynamics of single relativistic stars *Phys. Rev.* D **65** 084024
[192] Shibata M 2003 Axisymmetric general relativistic hydrodynamics: long term evolution of neutron stars and stellar collapse to neutron stars and black holes *Phys. Rev.* D **67** 024033
[193] Yoshida S and Eriguchi Y 2001 Quasiradial modes of rotating stars in general relativity *Mon. Not. R. Astron. Soc.* **322** 389–96
[194] Thierfelder M, Bernuzzi S, Hilditch D, Bruegmann B and Rezzolla L 2011 The trumpet solution from spherical gravitational collapse with puncture gauges *Phys. Rev.* D **83** 064022
[195] Gowdy R 1971 Gravitational waves in closed universes *Phys. Rev. Lett.* **27** 826–9
[196] New K C, Watt K, Misner C W and Centrella J M 1998 Stable three level leapfrog integration in numerical relativity *Phys. Rev.* D **58** 064022
[197] Woosley S E and Bloom J S 2006 The supernova—gamma-ray burst connection *Ann. Rev. Astron. Astrophys.* **44** 507–56
[198] Burrows A, Dessart L, Livne E, Ott C D and Murphy J 2007 Simulations of magnetically-driven supernova and hypernova explosions in the context of rapid rotation *Astrophys. J.* **664** 416–34
[199] Palenzuela C, Lehner L, Reula O and Rezzolla L 2009 Beyond ideal MHD: towards a more realistic modeling of relativistic astrophysical plasmas *Mon. Not. R. Astron. Soc.* **394** 1727–40
[200] Del Zanna L, Zanotti O, Bucciantini N and Londrillo P 2007 ECHO: an Eulerian conservative high order scheme for general relativistic magnetohydrodynamics and magnetodynamics *Astron. Astrophys.* **473** 11–30
[201] Farris B D, Li T K, Liu Y T and Shapiro S L 2008 Relativistic radiation magnetohydrodynamics in dynamical spacetimes: numerical methods and tests *Phys. Rev.* D **78** 024023
[202] Babiuc M, Szilagyi B, Winicour J and Zlochower Y 2011 A characteristic extraction tool for gravitational waveforms *Phys. Rev.* D **84** 044057
[203] Reisswig C, Bishop N, Pollney D and Szilágyi B 2010 Characteristic extraction in numerical relativity: binary black hole merger waveforms at null infinity *Class. Quantum Grav.* **27** 075014
[204] Tóth G 2000 The $\nabla \cdot B = 0$ constraint in shock-capturing magnetohydrodynamics codes *J. Comput. Phys.* **161** 605–52
[205] Tchekhovskoy A, McKinney J C and Narayan R 2007 WHAM: a WENO-based general relativistic numerical scheme I: hydrodynamics *Mon. Not. R. Astron. Soc.* **379** 469–97
[206] Zebrowski A, Löffler F and Schnetter E 2012 The BL-Octree: an efficient data structure for discretized block-based adaptive mesh refinement *Advances in Parallel Computing* vol 22 81–8