Abstract: Numerical relativity has undergone a revolution in the past decade. With a well-understood mathematical formalism, and full control over the gauge modes, it is now entering an era in which the science can be properly explored. In this work, we introduce GRChombo, a new numerical relativity code written to take full advantage of modern parallel computing techniques. GRChombo’s features include full adaptive mesh refinement with block structured Berger-Rigoutsos grid generation which supports non-trivial “many-boxes-in-many-boxes” meshing hierarchies, and massive parallelism through the Message Passing Interface (MPI). GRChombo evolves the Einstein equation with the standard BSSN formalism, with an option to turn on CCZ4 constraint damping if required. We show that GRChombo passes all the standard “Apples-to-Apples” code comparison tests. We also show that it can stably and accurately evolve vacuum black hole spacetimes such as binary black hole mergers, and non-vacuum spacetimes such as scalar collapses into black holes. As an illustration of its AMR capability, we demonstrate the evolution of triple black hole merger, which can be set up trivially in GRChombo.
Contents

1. Introduction 2

2. GRChombo 4
   2.1 GRChombo Adaptive-Mesh Finite-Difference Scheme 4
      2.1.1 Discretization and Time-stepping 4
      2.1.2 Berger-Rigoutsos Block-structured AMR 5
      2.1.3 Load Balancing 7
   2.2 GRChombo evolution equations 8
      2.2.1 Scalar Matter Evolution Equations 12
      2.2.2 Kreiss-Oliger Dissipation 13
      2.2.3 Boundary Conditions 13
      2.2.4 Initial conditions 13

3. Apples with Apples Tests 14
   3.1 Robust stability test 14
   3.2 Linear wave test 15
   3.3 Gauge wave tests 15
   3.4 Gowdy wave test 15

4. Vacuum black hole solutions 16
   4.1 Schwarzschild black hole 18
   4.2 Kerr black hole 18
   4.3 Boosted black hole 19
   4.4 Binary inspiral 20
   4.5 Triple black hole inspiral 20

5. Choptuik scalar field collapse 21

6. Discussion 21

A. Plots from code tests 27
1. Introduction

Almost a hundred years after Einstein wrote down the equations of General Relativity [1], solutions of the Einstein equation remain notoriously difficult to find beyond those which exhibit significant symmetries. Even for these highly symmetric solutions, basic questions remain unanswered. A famous example is the question of the non-perturbative stability of the Kerr solution – more than 50 years after its discovery, it is not fully known whether the exterior Kerr solution is stable. The main difficulty of solving the Einstein equation is its non-linearity, which defies perturbative approaches.

One of the main approaches in our hunt for solutions is the use of numerical methods. Numerical methods have been used to solve the Einstein equation for many decades, but the past decade has seen tremendous advances. A particular watershed moment was the breakthrough in evolving the inspiral mergers of two black holes [2–4] in 2005, a crucial milestone in the growth of numerical relativity as a discipline and as a tool. The other driver of this development is an explosion in the availability of large and powerful supercomputing clusters and the maturity of parallel processing technology such as the Message Passing Interface (MPI) and Open Multi-Processing (OpenMP) – which open up new computational approaches to solving the Einstein equation.

In the next decade or so, we expect this development to continue to accelerate. The gravitational wave detector LIGO is expected to start Advanced LIGO science runs late in 2015, and there are hopes that the sensitivity might be good enough to achieve a first detection of gravitational waves from binaries. In the longer term, the European Space Agency ESA has designated the space-based eLISA detector an L3 launch slot (expected launch date around 2034), and the LISA Pathfinder spacecraft has a firm launch date of late 2015.

Beyond searching for gravitational waves and black holes, numerical relativity is now beginning to find uses in the investigation of other areas of fundamental physics. For example, standard GR codes are now being adapted to study modified gravity [5], cosmology [6, 7] and even string theory [8–10]. In particular, there is increasing focus on solving strong-field coupled GR-matter equation – cosmic string evolution with GR, realistic black hole systems with accretion disks, non-perturbative systems in the early universe, etc. This nascent, but growing, interest in using numerical relativity as a mature scientific tool to explore other broad areas of physics is one key motivation of this work.

In many cases, numerical GR codes are specialized “one-offs”, written to solve a particular problem where some physical feature such as ellipticity and symmetries are used to simplify the solutions. Nevertheless, there exists several generalized multipurpose codes. Many well-known multipurpose GR codes have been built on top of
the CACTUS framework [11] – two publicly available implementations are the McLachlan/Kranc code [12,13], which is a finite difference code using the Baumgarte-Shapiro-Shibata-Nakagawa (BSSN) scheme [14,15], and the Whisky code, which is a numerical relativity with Magneto-Hydrodynamics (MHD) code [16]. A pseudo spectral code with harmonic gauge is implemented by the SPeC code [17].

In this work, we introduce GRChombo, a new multi-purpose numerical relativity code. GRChombo is built on top of the Chombo [18] framework. Chombo is a set of tools developed by Lawrence Berkeley National Laboratory for implementing block-structured adaptive mesh refinement (AMR) for solving partial differential equations. GRChombo features include the following.

- **BSSN formalism with moving puncture**: GRChombo evolves the Einstein equation in the BSSN formalism. An option to turn on the CCZ4 constraint damping modification [19, 20] is also available. Singularities of black holes are managed using the moving puncture gauge conditions [4].

- **Adaptive Mesh Refinement**: GRChombo supports full adaptive mesh refinement with non-trivial nesting topologies using standard Chombo libraries, via the Berger-Rigoutsos block-structured adaptive mesh algorithm [21]. The user only needs to specify regridding criteria, and GRChombo does the rest. Kreiss-Oliger dissipation is used to control errors, from both truncation and the interpolation associated with regridding.

- **MPI scalability**: GRChombo inherits the parallel infrastructure of Chombo, with ability to scale efficiently to many thousands of CPU-cores per run.

- **Standardized Output and Visualization**: GRChombo uses Chombo’s HDF5 output format, which is supported by many popular visualization tools such as VisIt. In particular, the output files can be used as input files if one chooses to continue a previously stopped run – i.e. the output files are also checkpoint files.

In this paper, we will detail the capabilities of GRChombo – its design methodology and its performance in the so-called “Apples with Apples” tests [22]. We will also present the results of several standard simulations. The paper is organized as follows.

In Sec. 2.1, we will describe the code itself, and in particular the AMR methodology. In Sec. 3 we detail the results of the standard Apples with Apples tests [22]. In Sec. 4 we

---

1Since the Chombo core is dimension-independent for up to six spatial dimensions, GRChombo could potentially be extended to simulate fully higher dimensional spacetimes without any symmetry assumptions, subject to computational resource availability.
demonstrate the ability of the code to stably evolve spacetimes containing black-hole-type singularities, the AMR capabilities of the code and its robustness to regridding errors. In Sec. 5 we demonstrate the ability of the code to evolve scalar fields with gravity, by recreating the results of the sub-critical and critical cases of Choptuik scalar field collapse detailed in [23]. We discuss our results and future directions in Sec. 6.

Videos of some of the tests, in particular the Choptuik collapse and binary and triple black hole mergers, can be viewed via our website at http://grchombo.github.io.

2. GRChombo

In this section, we will discuss our numerical implementation of the Einstein equation in GRChombo.

2.1 GRChombo Adaptive-Mesh Finite-Difference Scheme

GRChombo is a physics engine built around the publicly-available adaptive-mesh framework Chombo [18]. GRChombo solves the system of hyperbolic (see section 2.2) partial differential equations of the Einstein equation and scalar matter content using a finite difference scheme.

A key feature of GRChombo is its highly flexible adaptive mesh refinement capability – to be precise, GRChombo implements Berger-Oliger style [24,25] adaptive mesh refinement with Berger-Rigoutsos [21] block-structured grid generation. GRChombo supports full non-trivial mesh topology – i.e. many-boxes-in-many-boxes. Morton ordering is used to map grid responsibility to neighbouring processors in order to optimize processor number scaling.

2.1.1 Discretization and Time-stepping

We would like to evolve a set of fields in space (the state-vector) \( \Phi(x,t) = \{\phi_1, \phi_2, \phi_3, \ldots\} \) through time \( t \) via the equations of motion

\[
\frac{\partial \Phi}{\partial t} = \mathcal{F}(\Phi)
\]

(2.1)

where \( \mathcal{F} \) is some operator on \( \Phi \) which, in the case of the Einstein equation, is non-linear. In GRChombo, both the space and time coordinates are discretized. Evolution in time is achieved through time-stepping \( t \rightarrow t + \Delta t \), where at each time step we compute the fluxes for each grid point individually. Time stepping is implemented using the standard 4th Order Runge-Kutta method, and hence, as usual, we only need to store the values of the state-vector at each time step.
Φ itself is discretized into a cell-centered grid. Spatial derivatives across grid points are computed using standard 4-th order stenciling for all spatial derivatives, except for advection terms which are implemented using an upwind stencil [26].

2.1.2 Berger-Rigoutsos Block-structured AMR

GRChombo implements the Berger-Rigoutsos adaptive-mesh-refinement algorithm [21], which is one of the standard block-structured AMR schemes. Block-structured AMR regrids by overlaying variable size boxes, instead of remeshing on a cell-by-cell basis (the “bottom-up” approach). The main challenge is to find an efficient algorithm to partition the cells which need regridding into rectangular “blocks”. In this section, we will briefly discuss the algorithm.

For a given grid at some refinement level \( l \) where \( l = 0 \) is the base level and \( l_{\text{max}} \) is some preset maximum refinement level, we first “tag” cells for which refining is required. The refinement condition used by GRChombo is discussed later in this section. The primary problem of AMR is to efficiently partition this grid into regions which require adaptive remeshing. In block-structured AMR these regions are boxes in 3D or rectangles in 2D. Efficiency is measured by the ratio of tagged over untagged cell points in the final partitions.

In each partition, we compute the signatures or traces of the tagging function \( f(x, y, z) \) of any given box

\[
X(x) = \int f(x, y, z) dydz,
\]

\[
Y(y) = \int f(x, y, z) dx dz,
\]

\[
Z(z) = \int f(x, y, z) dy dx
\]

where \( f(x, y, z) = 1 \) if it is tagged for refinement and 0 otherwise. Given these traces, we can further compute the Laplacian of the traces \( \partial_x^2 X(x) \), \( \partial_y^2 Y(y) \) and \( \partial_z^2 Z(z) \). Given the Laplacians, the algorithm can search for all (if any) inflection points individually for each direction – i.e. the locations of zero crossings of the Laplacian, and then pick the one whose \( \delta(\partial_i^2 X_i) \) is the greatest (corresponding to the line – or plane in 3D – separating the largest change in the Laplacian). This point then becomes the line of partition for this particular dimension. Roughly speaking, this line corresponds to an edge between tagged and untagged cells in the orthogonal directions of the signature. Furthermore, if there exists a point \( x_i \) with zero signature \( X_i(x_i) = 0 \) (i.e. no cells
tagged along the plane orthogonal to the direction), then this “hole” is chosen to be the line of partition instead.

After a partitioning, we check whether or not each partition is efficient, specifically whether it passes a user-specified threshold or fill factor, $\epsilon < 1.0$,

$$\frac{\text{Tagged Cells}}{\text{Total Cells}} > \epsilon$$

(2.6)

If this is true, then we check if this box is properly nested\(^2\) [24, 25] and if so we accept this partition and the partitioning for this particular box stops. If not, then we continue to partition this box recursively until either all boxes are accepted or partitioning no longer can be achieved (either by the lack of any tagged cells or reaching a preset limit on the number of partitions). Furthermore, GRChombo allows one to set the maximum partition size, which if exceeded will force a partitioning of the box.

Note that a higher value of $\epsilon$ means that the partitioning will be more aggressive which will lead to a higher efficiency in terms of final ratio of tagged to untagged cells – generating more boxes in the process. However, this is not necessarily always computationally better as partitioning requires computational overhead, which depends on the number and topology of the processors. The ideal fill ratio is often a function of available processors, their topology and of course the physical problem in question.

A partitioned box is then refined, i.e. its grids split into a finer mesh using the (user definable) refinement ratio $n^l = \delta x^{l+1} / \delta x^l$, and this process continues recursively until we either have no more tagged cells, or when we reached a preset number of refinement level $l_{\text{max}}$.

Finally we need to specify a prescription for tagging which cells are required to be refined. GRChombo tags a cell when any (set of) user selected fields $\phi \in \Phi$ passes a chosen gradient threshold $\sigma(\phi)$, i.e.

$$f(x, y, z) = \begin{cases} 1 & \text{if } \partial_i \phi > \sigma(\phi) \text{ for any } i \\ 0 & \text{otherwise}. \end{cases}$$

(2.7)

This condition can be augmented, for example by using estimated truncation errors as tagging conditions instead.

Partitioning can be done at every time-step for each refinement level and this is a user preset choice per refinement level. However, the user may wish to select a

\(^2\)Properly nested means that (1) a $l+1$ level cell must be separated from an $l-1$ cell by at least a single $l$ level cell and (2) the physical region corresponding to a $l-1$ level cell must be completely filled by $l$ cells if it is refined, or it is completely unrefined (i.e. there cannot be “half-refined” coarse cells).
lower frequency because it might be useful to not partition at every timestep for a given refinement level. One consideration is that it is important to let numerical errors dissipate (e.g. via Kreiss-Oliger dissipation, see Sec. 2.2.2) before remeshing. Once a new hierarchy of partitions is determined, we interpolate via linear interpolation from coarse to fine mesh, and average from fine to coarse mesh.

Since the finer mesh has a smaller Courant number, each mesh level’s timestep is appropriately reduced via

$$\Delta t^{l+1} = \frac{\Delta t^l}{n^l}. \quad (2.8)$$

GRChombo follows standard Berger-Collela AMR evolution algorithm [25]. Starting from the coarsest mesh, it advances the coarse mesh 1 time step i.e. $t \rightarrow t + \Delta t^l$. Then it advances the next finest mesh $n^l$ times until the fine mesh “catches up” with the coarse mesh time. Once both coarse and fine mesh are at the same time $t$, GRChombo synchronizes them by averaging over the fine cells to the coarse cells values. We add that in a conservative system, this simple synchronization is not conservative and requires proper refluxing – the coarse fluxes are replaced with a time-averaged fine mesh fluxes. This step incurs additional overhead, and is at the moment not implemented by GRChombo as GR equations are not conservative. Nevertheless, we intend to implement conservative refluxing as an option in a future version of GRChombo.

2.1.3 Load Balancing

GRChombo’s efficiency when running on a large number of distributed-memory nodes is highly dependent on efficient load balancing of the available computational work across those nodes. Load balancing seeks to avoid the situation where most of the nodes are waiting for some small subset of nodes to finish their computational work, and it does this by seeking to distribute the amount of work to be done per time step evenly among all of the nodes. This can be non-trivial when AMR boxes at many different refinement levels are simultaneously being evolved across the system. In addition, even within a single node, multiple OpenMP threads might be running, and the per-node workload needs to be balanced amongst those threads.

For the inter-node load balancing, GRChombo leverages Chombo’s load balancing capabilities to distribute the AMR boxes among the available nodes. It does this by building a graph of the boxes to be distributed, adding edges between neighbouring and overlapping boxes. A bin packing / knapsack algorithm is used to balance the computational work among nodes, where the work is assumed to be proportional to the number of grid points, and then an exchange phase is used to minimise the communication cost. Because this load balancing procedure can be costly, we normally run it
only every few time steps. In between runs of the load balancing procedure, new boxes generated by AMR refinement stay on the node which holds the parent box.

Within each node, the computational work is divided amongst the available OpenMP threads by iterating over the boxes to process using OpenMP’s dynamic scheduling capability. This allows each thread to take the next available box from the queue of unprocessed boxes, instead of deciding ahead of time which boxes each thread will process. This is important because the boxes are varying in size. We generally divide even the coarsest level into multiple boxes so that it can be processed in parallel by multiple threads.

2.2 GRChombo evolution equations

Many of the numerical relativity codes implement the so called BSSN formulation of the Einstein equation [14,15,27]. This formulation expresses the Einstein equation in a strongly hyperbolic form, and together with the “1+log” slicing [28] and the “gamma-driver” gauge conditions [29], have allowed us to stably simulate dynamical spacetimes of interest, including black hole binaries. More recently, other refined formulations of the Einstein equation based on the Z4 system [3, 20] have been proposed, most notably the Z4c formulation [30] and the CCZ4 formulation [19].\(^3\) In the Z4 system, both the Hamiltonian and the momentum constraint are promoted to become dynamical variables and hence constraint violating modes can propagate and eventually exit the computational domain. This may potentially results in more a stable evolution. In addition, the Z4 system can be augmented with damping terms so that constraint violating modes can be exponentially suppressed. In practical terms, the changes required between the CCZ4 equations and the standard BSSN equations are minimal and in GRChombo we have implemented both.

In this work, we follow the indexing convention of [34]. The signature is \((-+++)\), and low-counting Latin indices \(a, b, \ldots\) are abstract tensor indices while Greek indices \(\mu, \nu, \ldots\) denote spacetime component indices and run from 0, 1, 2, 3. Spatial component indices are labeled by high-counting Latin indices \(i, j, \ldots\) which runs from 1, 2, 3. Unless otherwise stated, we set \(G = 1\) and \(c = 1\).

The Z4 system with constraint damping is [3]

\[
R_{ab} + \nabla_a Z_b + \nabla_b Z_a - \kappa_1 \left[ n_a Z_b + n_b Z_a - (1 + \kappa_2) g_{ab} n^c Z_c \right] = 8 \pi \left( T_{ab} - \frac{1}{2} g_{ab} T \right) \tag{2.9}
\]

\(^3\)Both the BSSN and CCZ4 equations have been written in a fully covariant form [31–33]. These covariant formulations can be advantageous in certain cases, and we plan to implement them in the future.
where $R_{ab}$ is the Ricci tensor associated to the metric $g$ on the spacetime manifold $\mathcal{M}$, and $\nabla$ is the corresponding metric compatible covariant derivative. $T_{ab}$ is the stress-energy tensor of the matter and $T \equiv g_{ab} T^{ab}$ is its trace. If we set $Z^a = 0$, the Z4 equations Eqn. (2.9) reduce to the standard (trace-reversed) Einstein equation. Here $\kappa_1$ and $\kappa_2$ are constants that control the damping.

In the GRChombo code we use the standard $3 + 1$ ADM decomposition of the spacetime metric,

$$ds^2 = -\alpha^2 dt^2 + \gamma_{ij}(dx^i + \beta^i dt)(dx^j + \beta^j dt), \quad (2.10)$$

so that $\gamma_{ij}$ is the induced metric on the spatial slices and

$$n^\mu = \frac{1}{\alpha} \left( \partial^\mu - \beta^i \partial_i^\mu \right), \quad (2.11)$$

is the corresponding timelike unit normal. The extrinsic curvature is defined as

$$K_{ij} = -\frac{1}{2} (\mathcal{L}_n \gamma)_{ij}, \quad (2.12)$$

where $\mathcal{L}$ denotes the Lie derivative. As it is customary, we decompose the induced metric as $\gamma_{ij} = \frac{1}{\chi^2} \tilde{\gamma}_{ij}$ so that $\det \tilde{\gamma}_{ij} = 1$ and $\chi = (\det \gamma_{ij})^{-\frac{1}{6}}$. Similarly, the extrinsic curvature is decomposed into its trace, $K = \gamma^{ij} K_{ij}$, and its traceless part so that

$$K_{ij} = \frac{1}{\chi^2} \left( \tilde{A}_{ij} + \frac{1}{3} \tilde{K} \tilde{\gamma}_{ij} \right), \quad (2.13)$$

with $\tilde{\gamma}^{ij} \tilde{A}_{ij} = 0$. In the Z4 system, one further defines $\Theta$ as the projection of the Z4 four-vector along the normal timelike direction, $\Theta \equiv -n_\mu Z^\mu$. Finally, the spacelike components of the four-vector, $Z_i$, are included in a variable $\hat{\Gamma}^i$ defined as

$$\hat{\Gamma}^i \equiv \tilde{\Gamma}^i + 2 \tilde{\gamma}^{ij} Z_j, \quad (2.14)$$

where $\tilde{\Gamma}^i = \tilde{\gamma}^{jk} \tilde{\Gamma}^i_{jk}$ and $\hat{\Gamma}^i_{jk}$ are the Christoffel symbols associated to the conformal metric $\tilde{\gamma}_{ij}$,

$$\hat{\Gamma}^i_{jk} = \frac{1}{2} \tilde{\gamma}^{il} \left( \partial_j \tilde{\gamma}_{kl} + \partial_k \tilde{\gamma}_{jl} - \partial_l \tilde{\gamma}_{jk} \right). \quad (2.15)$$

Summarizing, the dynamical variables for the Z4 system are

$$\{ \chi, \tilde{\gamma}_{ij}, K, \tilde{A}_{ij}, \Theta, \hat{\Gamma}^i \}. \quad (2.16)$$

Setting to zero the Z4 four-vector, $Z^\mu = 0$, this system reduces to the standard BSSN system.

---

4We add that GRChombo does not impose the condition $\det \tilde{\gamma}_{ij} = 1$ by hand.
Finally, we recall the various components of the matter stress tensor in the standard $(3+1)$-decomposition:
\[
\rho = n_a n_b T^{ab}, \quad S_i = -\gamma_{ia} n_b T^{ab}, \quad S_{ij} = \gamma_{ia} \gamma_{jb} T^{ab}, \quad S = \gamma^{ij} S_{ij}.
\] (2.17)

We are now ready to write down the evolution equations for CCZ4 system in the standard $(3+1)$-decomposition [19]:
\[
\partial_t \chi = \frac{1}{3} \alpha \chi K - \frac{1}{3} \chi \partial_k \beta^k + \beta^k \partial_k \chi, \quad (2.18)
\]
\[
\partial_t \tilde{\gamma}_{ij} = -2 \alpha \tilde{A}_{ij} + \tilde{\gamma}_{ik} \partial_j \beta^k + \tilde{\gamma}_{jk} \partial_i \beta^k - \frac{2}{3} \tilde{\gamma}_{ij} \partial_k \beta^k + \beta^k \partial_k \tilde{\gamma}_{ij}, \quad (2.19)
\]
\[
\partial_t K = -\gamma^{ij} D_i D_j \alpha + \alpha \left( R + 2 D_i Z^i + K^2 - 2 K \Theta \right) + \beta^i \partial_i K
- 3 \alpha \kappa_1 (1 + \kappa_2) \Theta + 4 \pi \alpha (S - 3 \rho), \quad (2.20)
\]
\[
\partial_t \tilde{A}_{ij} = \chi^2 \left[ -D_i D_j \alpha + \alpha \left( R_{ij} + D_i Z_j + D_j Z_i - 8 \pi \alpha S_{ij} \right) \right]^{TF}
+ \alpha \tilde{A}_{ij} (K - 2 \Theta) - 2 \alpha \tilde{A}_{ii} \tilde{A}_{ij} + \tilde{A}_{ik} \partial_j \beta^k + \tilde{A}_{jk} \partial_i \beta^k
- \frac{2}{3} \tilde{A}_{ij} \partial_k \beta^k + \beta^k \partial_k \tilde{A}_{ij}, \quad (2.21)
\]
\[
\partial_t \Theta = \frac{1}{2} \alpha \left( R + 2 D_i Z^i - \tilde{A}_{ij} \tilde{A}^{ij} + \frac{2}{3} K^2 - 2 \Theta K \right)
- Z^i \partial_i \alpha + \beta^k \partial_k \Theta - \alpha \kappa_1 (2 + \kappa_2) \Theta - 8 \pi \alpha \rho, \quad (2.22)
\]
\[
\partial_t \tilde{\Gamma}^i = -2 \tilde{A}^{ij} \partial_j \alpha + 2 \alpha \left( \tilde{\Gamma}^i_{jk} \tilde{A}^{jk} - \frac{2}{3} \tilde{\gamma}^{ij} \partial_j K - 3 \tilde{A}^{ij} \frac{\partial_j \chi}{\chi} \right)
+ \beta^k \partial_k \tilde{\Gamma}^i + \tilde{\gamma}^{jk} \partial_j \partial_k \beta^i + \frac{1}{3} \tilde{\gamma}^{ij} \partial_{ij} \chi
+ \frac{2}{3} \tilde{\Gamma}^i \partial_k \beta^k - \tilde{\Gamma}^k \partial_k \beta^i + 2 \kappa_3 \left( \frac{2}{3} \tilde{\gamma}^{ij} Z_j \partial_k \beta^k - \tilde{\gamma}^{jk} Z_j \partial_k \beta^i \right)
+ 2 \tilde{\gamma}^{ij} \left( \alpha \partial_j \Theta - \Theta \partial_j \alpha - \frac{2}{3} \alpha K Z_j \right) - 2 \alpha \kappa_1 \tilde{\gamma}^{ij} Z_j - 16 \pi \alpha \tilde{\gamma}^{ij} S_j. \quad (2.23)
\]

Here $D_i$ is the metric compatible covariant derivative with respect to the physical metric $\gamma_{ij}$ and $[\ldots]^{TF}$ denotes the trace free part of the expression inside the parenthesis. The three-dimensional Ricci tensor, $R_{ij}$, is split as
\[
R_{ij} = \tilde{R}_{ij} + R_{ij}^\chi, \quad (2.24)
\]
where
\[
\tilde{R}_{ij} = -\frac{1}{2} \tilde{\gamma}^{lm} \partial_m \partial_l \tilde{\gamma}_{ij} + \tilde{\Gamma}^k \tilde{\Gamma}_{(ij)k} + \tilde{\gamma}^{lm} (2 \tilde{\Gamma}_{(i}^{k} \tilde{\Gamma}_{j)km} + \tilde{\Gamma}_{lm} \tilde{\Gamma}_{kl}) \quad (2.25)
\]
and
\[
R_{ij}^\chi = \frac{1}{\chi} (\tilde{D}_i \tilde{D}_j \chi + \tilde{\gamma}_{ij} \tilde{D}^l \tilde{D}_l \chi) - \frac{2}{\chi^2} \tilde{\gamma}_{ij} \tilde{D}^l \chi \tilde{D}_l \chi. \quad (2.26)
\]
where \( \tilde{D}_i \) is the metric compatible covariant derivative with respect to the conformal metric \( \tilde{\gamma}_{ij} \). Note that the three-dimensional Ricci Scalar is then \( R = \gamma^{ij} R_{ij} \).

To complete the set of evolution equations, we need to choose slicing conditions – we specify the gauge and their driving conditions for the lapse \( \alpha \) and shift \( \beta^i \) [29]. The optimal gauge conditions are in general physics dependent. \texttt{GRChombo} allows the user to handcode in any gauge conditions that is suited for the application at hand.

The \textit{alpha-driver} condition is usually written as a first order differential equation

\[
\partial_t \alpha = -\mu_{\alpha_1} \alpha^{\mu_2} K + \mu_{\alpha_3} \beta_i \partial_i \alpha. \tag{2.27}
\]

The commonly used 1 + log slicing applicable for black hole inspirals corresponds to \( \mu_{\alpha_1} = 1, \mu_{\alpha_2} = 1 \) and \( \mu_{\alpha_3} = 1 \). On the other hand, the \textit{maximal slicing} condition, which preserves \( K = 0 \) and \( \partial_t K = 0 \) at all slices, is a second order differential equation

\[
D^2 \alpha = \alpha [K_{ij} K^{ij} + 4\pi (\rho + S)], \tag{2.28}
\]

which is useful for spherically symmetric collapse problems such as the Choptuik scalar collapse scenarios.

We specify the evolution equation for \( \beta^i \) using the \textit{gamma-driver} conditions [29],

\[
\partial_t \beta^i = \eta_1 B^i \tag{2.29}
\]

\[
\partial_t B^i = \mu_{\beta_1} \alpha^{\mu_2} \partial_i \tilde{\Gamma} - \eta_2 B^i, \tag{2.30}
\]

where \( B^i \) is an auxiliary vector field, while \( \eta_1, \eta_2, \mu_{\beta_1} \) and \( \mu_{\beta_2} \) are input parameters.

The usual hyperbolic gamma-driver condition uses the parameters \( \eta_1 = 3/4, \mu_{\beta_1} = 1, \mu_{\beta_2} = 0 \) and \( \eta_2 = 1 \). We have also included parameters that allow us to turn on standard advection terms in Eqn. (2.29)–Eqn. (2.30). In Sec. 3, we do not have to deal with moving black holes and hence the gauges are simpler to specify in the tests – we will describe them in those sections. In Sec. 4 and Sec. 5, where black holes are present, we manage the singularities with the so-called \textit{moving punctures method} [4,35]. In addition, we hard code the condition \( \alpha > 0 \) as in usual practice.

Finally, \texttt{GRChombo} computes both the Hamiltonian constraint,

\[
H = R + K^2 - K_{ij} K^{ij} - 16\pi \rho \tag{2.31}
\]

and the momentum constraint,

\[
M_i = \gamma^{jk} (\partial_t K_{ij} - \partial_i K_{jl} - \Gamma^m_{jil} K_{mi} + \Gamma^m_{ij} K_{lm}) - 8\pi S_i. \tag{2.32}
\]

in order to monitor the accuracy of the calculation.
Equations Eqn. (2.18)–Eqn. (2.23) are the CCZ4 evolution equations as originally presented in [19], including the extra damping parameter \( \kappa_3 \). This parameter controls the coupling of some quadratic terms in the evolution equation for \( \hat{\Gamma}^i \). The choice \( \kappa_3 = 1 \) corresponds to the fully covariant CCZ4 system, but as discussed in [19], it leads to instabilities in the evolution of spacetimes containing black holes. More recently, [36] showed that replacing \( \kappa_1 \rightarrow \kappa_1/\alpha \) in Eqn. (2.18)–Eqn. (2.23) allows to stably evolve black hole spacetimes whilst retaining the full covariance of the CCZ4 system. In GRChombo we have included a parameter that allows us to switch from the original formulation of the CCZ4 system to the more recent one proposed in [36], with the aforementioned redefinition of \( \kappa_1 \).

Note that in the actual evolution, the values of the three-vector \( Z_i \) are computed from the knowledge of the evolved variable \( \hat{\Gamma}^i \) and \( \tilde{\Gamma}^i \), which is computed from the conformal metric, \( \tilde{\gamma}_{ij} \). Finally, we note that the evolution equations Eqn. (2.18)–Eqn. (2.23) reduce to the standard BSSN equations upon setting \( \Theta = 0 \) and \( Z^i = 0 \), and using the Hamiltonian constraint, Eqn. (2.31), in the evolution equation for \( K \), Eqn. (2.20), to eliminate the Ricci scalar \( R \).

### 2.2.1 Scalar Matter Evolution Equations

In this work, we included a single minimally coupled scalar field \( \phi \) as matter content

\[
L_\phi = \frac{1}{2} \nabla_\mu \phi \nabla^\mu \phi + V(\phi),
\]

with the equation of motion

\[
\nabla_\mu \nabla^\mu \phi - \frac{dV}{d\phi} = 0.
\]

As usual, we decompose the second order Eqn. (2.34) into two first order variables \( \phi \) and \( \Pi_M \)

\[
\Pi_M \equiv \frac{1}{\alpha} (\partial_t \phi - \beta^i \partial_i \phi).
\]

We note that our \( \Pi_M \) is negative of \( \Pi \) in some references, e.g. [34]. Eqn. (2.34) is then decomposed into the following equations

\[
\partial_t \phi = \alpha \Pi_M + \beta^i \partial_i \phi
\]

and

\[
\partial_i \Pi_M = \beta^j \partial_i \Pi_M + \gamma^{ij} (\alpha \partial_j \partial_i \phi + \partial_j \phi \partial_i \alpha) + \alpha \left( K \Pi_M - \gamma^{ij} \Gamma^k_{ij} \partial_k \phi + \frac{dV}{d\phi} \right).
\]

We also use the energy momentum tensor of the scalar field

\[
T_{\mu\nu} = \nabla_\mu \phi \nabla_\nu \phi - \frac{1}{2} g_{\mu\nu} (\nabla_\lambda \phi \nabla^\lambda \phi + 2V)
\]

to calculate the matter components of the BSSN/CCZ4 system via the Eqn. (2.17).
2.2.2 Kreiss-Oliger Dissipation

In a finite difference scheme, instabilities can arise from the appearance of high frequency spurious modes. Furthermore, regridding generates errors an order higher than the typical error of the evolution operator, hence it is doubly crucial that we control these errors. The standard prescription to deal with this is to implement some form of numerical dissipation to damp out these modes. GRChombo implements \( N = 3 \) Kreiss-Oliger [37] dissipation. In this scheme, for all evolution variables \( u \in \{ \tilde{A}_{ij}, \tilde{\gamma}_{ij}, K, \chi, \Theta, \Gamma^i \} \), the evolution equations are modified as follows

\[
\partial_t u_m \rightarrow \partial_t u_m + \frac{\sigma}{64\Delta x} \left( u_{m+3} - 6u_{m+2} + 15u_{m+1} - 20u_m + 15u_{m-1} - 6u_{m-2} + u_{m-3} \right),
\]

where \( m \pm n \) labels the grid point \( m \), \( n \) the total offset from \( m \) and \( \sigma \) is an adjustable dissipation parameter usually of the order \( \mathcal{O}(10^{-2}) \). This 3rd order scheme is accurate as long as the integration order of the finite difference scheme is 5 or less (which it is in our implementation using 4th order Runge-Kutta).

2.2.3 Boundary Conditions

GRChombo supports both periodic (in any direction) boundary conditions, as well as any particular boundary conditions the user may want to specify (such as Neumann or Dirichlet types). A particular popular type of boundary condition is the so-called Sommerfeld [29] boundary condition, where out-going radiation are dissipated away. For any field \( f \), we impose the condition at the boundary

\[
\frac{\partial f}{\partial t} = \frac{v x_i}{r} \frac{\partial f}{\partial x_i} - \frac{v f - f_0}{r}
\]

where \( r = \sqrt{x_1^2 + x_2^2 + x_3^2} \) is the radial distance from the center of the grid, \( f_0 \) is the desired space-time at the boundary (typically Minkowski space for asymptotically flat spacetimes) and \( v \) the velocity of the “radiation”, which is typically chosen to be 1.

2.2.4 Initial conditions

GRChombo supports several ways of entering initial conditions.

- Direct equations – Initial conditions which are described by known analytic equations, such as the Schwarzschild solution, can be entered directly in equations form.

- Checkpointing – The HDF5 format output files from GRChombo doubles as checkpointing files. A run can simply be continued from any previous state as long as its HDF5 output file is available.
• Entering from data file – GRChombo allows one to insert data from a file.

• Relaxation – GRChombo has a rudimentary capability to solve for the initial metric given some initial mass distribution. Given a guess metric, GRChombo relaxes it to the correct initial metric using a dissipation term which is proportional to a user chosen dissipation coefficient times the Hamiltonian constraint.

The initial conditions used in this paper are mostly analytic or approximate analytic solutions, and so are entered directly into the code. In the Choptui collapse, a Mathematica numerical solution as a function of the radius is interpolated onto the initial grid.

3. Apples with Apples Tests

In this section we describe the results of applying the code to the standard Apples with Apples tests in [22]. Here we give a brief description of the key features of the tests, but the reader should refer to the paper for full specifications. Where we do not specify details, our treatment can be assumed to follow that of the standard tests.

The AMR capabilities of the code are not utilised in these tests, which are run at fixed resolutions, in order to make our results comparable to other codes. We consider the effects of regridding on code performance in Section 4.

All the test figures referred to in this section can be found in Appendix A.

3.1 Robust stability test

The robust stability test introduces small amounts of random noise, scaled with the grid spacing, to all of the evolution variables, in order to test the code’s robustness against numerical noise.

The test was conducted at resolutions of $\rho = 4$, $\rho = 2$ and $\rho = 1$, corresponding to grid spacings of 0.005, 0.01 and 0.02 respectively. No dissipation was added in the test.

It was seen that error growth in the evolution variables did not increase with increasing grid resolution, and similarly that the Hamiltonian constraint $H$ did not grow more for higher resolutions, as demonstrated in Figure 7, meaning that the test was passed.

The test was repeated with the CCZ4 system, as shown in Figure 8. The test is again passed, and it is seen that CCZ4 suppresses the Hamiltonian constraint, as expected.
3.2 Linear wave test

A wave of fixed amplitude is propagated across the grid in the $x$ direction with periodic boundary conditions. The amplitude is small enough that the non-linear terms are below numerical precision, such that the behaviour under the Einstein equation is approximately linear.

The test measures the errors in magnitude and phase introduced by the code after 1000 crossing times. As can be seen from Figure 9, there is negligible error in magnitude and phase after the 1000 crossing times. The performance is significantly better than that found for the other codes detailed in [22], in which the errors in phase were visible by eye.

3.3 Gauge wave tests

The BSSN formulation is known to produce unsatisfactory results for the gauge wave tests. GRChombo is no different in this respect. As can be seen in Figure 10, it becomes unstable after around 50 crossing times, with the Hamiltonian constraint increasing exponentially, even for a relatively small initial amplitude of the gauge wave of $A = 0.1$.

We used the harmonic slicing condition

$$\partial_t \alpha = -\alpha^2 K$$

(3.1)

to evolve the lapse and did not evolve the shift, following the treatment in [19]. We used a Kreiss-Oliger dissipation coefficient of $\sigma = 0.1$. A longer evolution can be achieved by adding more dissipation but long term stability cannot in general be achieved with a simple BSSN formulation.

It is possible to achieve stability by adding in the CCZ4 constraint damping terms, as first suggested in [19]. Turning on the constraint damping term in GRChombo with $A = 0.1$, we achieve a stable evolution, as shown in Figure 10, as expected.

3.4 Gowdy wave test

The Gowdy wave evolves a strongly curved spacetime: an expanding vacuum universe containing a plane polarised gravitational wave propagating around a 3-torus.

We used the analytic gauge to evolve the spacetime in the expanding direction, as described in [22], using the fact that:

$$\partial_t \alpha = -\frac{\partial_t \gamma_{zz}}{2\gamma_{zz}^{1/2}}$$

(3.2)

We evolve the collapsing direction starting the simulation at $t = t_0$ with the harmonic slicing described in [22] for the lapse, and zero shift, and evolving with the harmonic slicing condition Eqn. (3.1).
A Kreiss-Oliger dissipation coefficient of $\sigma = 0.05$ was used in both directions. The results for BSSN in the collapsing direction are shown in figure 11, and in the expanding direction in figure 12.

As is found in the Apples with Apples tests [22] for other simple BSSN codes, and as for the gauge wave tests, GRChombo with BSSN gives a less than satisfactory performance in this test in the expanding direction. The evolution is stable for approximately the first 30 crossing times, after which high frequency instabilities develop and cause code crash, due to the exponentially growing $\gamma_{zz}$ component. In [22] it was found that this behaviour, which occurred in the LazEvBSSN code, could be controlled with dissipation, but that long term accuracy was not achievable. The results for CCZ4 are presented as well for comparison, but this formulation also does not give a satisfactory evolution in the Apples with Apples gauge choice.

In the contracting direction the evolution is stable for the full 1000 crossing times and the test is passed. We were also able to test the 4th order convergence of the code, as shown in Figure 13.

4. Vacuum black hole solutions

In this section we show that the code can stably evolve spacetimes containing black hole type singularities. We also utilise the AMR abilities of the code and show that increasing the resolution does not result in significantly increased constraint violation or instability. All the simulations used the BSSN formulation, along with the gamma-driver and alpha-driver gauge conditions, and a Kreiss-Oliger dissipation coefficient of approximately $\sigma = 0.04$. Adding CCZ4 constraint damping gives better performance for the Hamiltonian constraint, as would be expected, but the results are broadly similar and so are not presented here.

All the test figures referred to in this section can be found in Appendix A.

The simulations are performed at two resolutions. At the lower resolution a regridding threshold of the gradient in $\chi$ of 0.01 is used, with a maximum of 7 grids, with the resolution on the top grid being 0.0625M. At the higher resolution a threshold of 0.005 is used, and a maximum of 8 grids and top resolution of 0.003125M. The difference this makes to the regridding is illustrated in Figure 1.

We use a simple apparent horizon finder to confirm that apparent horizons have formed. This calculates the value of the expansion of the future outgoing null geodesic, as in [38], at each gridpoint:

$$\Theta = \nabla_i n^i + K_{ij} n^i n^j - K$$  \hspace{1cm} (4.1)
Our horizon finder assumes approximate spherical symmetry, and an initial guess for the centre of the apparent horizon must be provided. This allows the values of $\Theta$ to be directly plotted across the grid. Where $\Theta = 0$, this indicates that an apparent
horizon has formed.

4.1 Schwarzschild black hole

First we evolve a standard Schwarzschild black hole, in the isotropic gauge, with a conformally flat metric, the lapse initially set to one everywhere, and the conformal factor $\chi$:

$$\chi = \left(1 + \frac{M}{2r}\right)^{-2} \quad (4.2)$$

The initial value of $\chi$ through a slice is shown in figure 2. We see the expected “collapse of the lapse” at the singularity and the solution quickly stabilises into the “trumpet” puncture solution described in [39]. We find an apparent horizon and are able to evolve the black hole stably and without code crash for over $t = 500M$ time steps.

We monitor the ADM mass of the black hole by integrating over a surface near the asymptotically flat boundary. In figure 14, it can be seen that this measure of the mass remains stable over more than $t = 100M$ time steps. We also monitor the angular momentum and linear momentum of the black hole, and find that these remain zero as expected, as shown in figure 14. These simple ADM measures rely on asymptotic flatness at the surface over which they are integrated, and so are sensitive to errors introduced by reflections at the boundaries, initial transients from approximate gauge choice or if the black hole is moving nearer the boundary (as in the boosted case). They are therefore less reliable as the simulation progresses, but we do confirm that we are evolving the correct spacetime.

The $L_\infty$ norm of the Hamiltonian constraint is shown in figure 15 at three times during the first $t = 100M$ steps. It is seen that, although there errors are relatively large at the grid boundaries (as a result of regridding and grid boundary errors), these are stable over the simulation time, and in fact decrease as the simulation progresses.

The simulation is repeated with the higher resolution (i.e., with 8 grids, as described above), and we see that although the new innermost grid introduces new errors at its boundary, the results are consistent with those from the lower resolution, and the solution is not destabilised.

4.2 Kerr black hole

We evolve a Kerr spacetime described by the quasi isotropic gauge described in [40] with the ratio $a = J/M$ set to 0.2. The ADM measures for the three components of the angular momenta and the mass are shown in figure 16, for the two resolutions described above.
Figure 2: The profile for chi through a slice perpendicular to the z axis is shown.

(a) Initial position  (b) Position at $t = 100$  (c) Grid at $t = 100$

Figure 3: Boosted black hole, movement: The boosted black hole moves across the grid diagonally with initial momenta of $P_x = 0.02$, $P_y = 0.02$ and $P_z = 0.0$, as expected, and the grid adapts to this movement, with the high resolution grids following the movement.

The time evolution of the $L_\infty$ norm of the Hamiltonian constraint is shown in figure 17. It is seen that this is stable over the simulation time.

4.3 Boosted black hole

We evolve a boosted black hole using the perturbative approximation from [34], with initial momenta set to $P_x = 0.02$, $P_y = 0.02$ and $P_z = 0.0$. The black hole moves across the grid diagonally, as is seen in Figure 3.
Figure 4: Binary Black Hole merger: Two black holes are evolved with GRChombo. The final stages of the merger are shown.

The ADM measures for the momenta are shown in figure 18, for the two resolutions. The time evolution of the $L_\infty$ norm of the Hamiltonian constraint is shown in figure 19. It is seen that this is stable over the simulation time.

4.4 Binary inspiral

We superpose the initial perturbative solution for two boosted black holes in [34], sufficiently separated, to simulate a binary inspiral merger.

We are able to evolve the merger stably such that the two black holes merge to form one with a mass approximately equal to the sum of the two. The progression of the merger is shown in figure 4. Analysis of the gravitational waves emitted from such a merger will be analysed in a future publication.

4.5 Triple black hole inspiral

The AMR abilities of the code allow us to easily evolve less symmetric spacetimes than simple black holes or binary mergers. The mesh adaptation in a triple black hole merger
Figure 5: Triple Black Hole merger: Three black holes are being evolved with GRChombo. The mesh is shown, which has adapted to the local curvature in $\chi$, the variable plotted.

is shown in figure 5. Videos of this merger, and the binary one, can be viewed on our website at http://grchombo.github.io.

5. Choptuik scalar field collapse

We now test the scalar field part of the code, by simulating the Choptuik scalar field collapse as described in [23] and illustrated in figure 6. The referenced description is for a 1+1 simulation which is evolved using a constrained evolution, such that the lapse $\alpha$ and the single degree of freedom for the metric, $A$, are both solved for on each slice using ODEs obtained from the constraint equations. The only degrees of freedom which are truly evolved are those of the field, $\phi$, $\Psi$ and $\Pi$.

Our evolution is carried out using the full 3+1 BSSN equations, without assuming or adapting coordinates to spherical symmetry. We are able to replicate the results obtained in [23], subject to some minor differences due to the fact that we evolve with the puncture gauge rather than according to the maximal slicing constraint equation, see figures 20 and 21, which can be found in Appendix A. Videos of the results can be viewed via our website at http://grchombo.github.io.

We see that GRChombo can accurately evolve the field profile in the presence of gravity, and copes with the collapse of the supercritical case into a singularity, without code crash. For the subcritical cases we see that the field disperses as expected.

6. Discussion

In this paper, we introduced and described GRChombo, a new multi-purpose numerical relativity code built using the Chombo framework. It is a 3+1D finite difference code
Figure 6: In Choptuik scalar field collapse, the initial specially symmetric configuration in the first figure (which shows the values on a slice perpendicular to the z axis) collapses, splitting into an ingoing and an outgoing wave as seen in the second image. If the amplitude of the initial perturbation is greater than a certain critical value, the ingoing wave will result in the formation of a black hole, as seen from the output of the apparent horizon finder in the third figure, which shows that an apparent horizon with a mass of about 0.25 has formed by $t = 15$.

based on the BSSN/CCZ4 evolution scheme. It supports Berger-Collela type AMR evolution with Berger-Rigoutsos block structured grid generation, and is fully parallelized via the Message Passing Interface, and time evolution is via standard 4-th order Runge-Kutta time-stepping.

We showed that it successfully passes the standard “Apples to Apples” tests. In addition to these tests, we evolved standard single black hole spacetimes (Schwarzschild and Kerr) and showed that it is stable for more than a thousand light crossing times. Using the moving puncture gauge, we also show that GRChombo stably evolve the merger of two and three black holes. We emphasise that setting the initial conditions for these mergers are trivial – GRChombo automatically remeshes the grid given a set of analytic initial conditions without further user intervention. Finally, we simulated the supercritical collapse of a scalar field configuration, and found that it forms a black hole as expected.

Nevertheless, despite its power, the AMR capability of GRChombo has to be treated with care. As we mentioned earlier, coarse-fine boundaries could be a significant source of inaccuracy, even though the Hamiltonian constraint may still be kept under control. A way to reduce coarse-fine boundary errors is to introduce conservative refluxing during interlevel operations. Although refluxing requires significant overhead, we intend

---

5We note that we perform as well as any other BSSN code in the Gowdy wave test as we expected in a pre-determined gauge. Although, [41] managed to achieve long-term evolution by considering different gauge conditions.
to implement it in the next iteration of \texttt{GRChombo}. The litmus test for accuracy of \texttt{GRChombo} is its ability to make accurate predictions of outgoing gravitational waveforms. We leave this, and the introduction of a set of “best practices” for the use of AMR in general relativistic systems, for a follow-up work.

\section*{Acknowledgements}

We would like to thank Erik Schnetter, Ulrich Sperhake, Helvi Witek, Luis Lehner, Carlos Palenzuela and Tom Giblin for many useful conversations. We would especially like to thank Juha Jäykkä and James Briggs for their amazing technical support. This work was undertaken on the COSMOS Shared Memory system at DAMTP, University of Cambridge operated on behalf of the STFC DiRAC HPC Facility. This equipment is funded by BIS National E-infrastructure capital grant ST/J005673/1 and STFC grants ST/H008586/1, ST/K00333X/1. EAL acknowledges support from an STFC AGP grant ST/L000717/1. PF and ST are supported by the European Research Council grant ERC-2011-StG279363HiDGR. PF is also supported by the Stephen Hawking Advanced Research Fellowship from the Centre for Theoretical Cosmology, University of Cambridge. MK is supported by an STFC studentship. He started his work on this project as a summer student funded by the Bridgewater Summer Undergraduate Research Programme at the Centre for Mathematical Sciences, University of Cambridge, and by King’s College, Cambridge. HF is supported by the US Department of Energy (DOE), and this research used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility, both supported under Contract DE-AC02-06CH11357.

\section*{References}

\begin{enumerate}
\item A. Einstein, “Die Grundlage der allgemeinen Relativitätstheorie,” \textit{Annalen der Physik} \textbf{354} (1916) no.7, 769–822.
\item F. Pretorius, “Evolution of binary black hole spacetimes,” \textit{Phys.Rev.Lett.} \textbf{95} (2005) 121101, \texttt{arXiv:gr-qc/0507014 [gr-qc]}.
\item C. Gundlach, J. M. Martin-Garcia, G. Calabrese, and I. Hinder, “Constraint damping in the Z4 formulation and harmonic gauge,” \textit{Class.Quant.Grav.} \textbf{22} (2005) 3767–3774, \texttt{arXiv:gr-qc/0504114 [gr-qc]}.
\item M. Campanelli, C. Lousto, P. Marronetti, and Y. Zlochower, “Accurate evolutions of orbiting black-hole binaries without excision,” \textit{Phys.Rev.Lett.} \textbf{96} (2006) 111101, \texttt{arXiv:gr-qc/0511048 [gr-qc]}.
\end{enumerate}
[5] E. Berti, E. Barausse, V. Cardoso, L. Gualtieri, P. Pani, et al., “Testing General Relativity with Present and Future Astrophysical Observations,” arXiv:1501.07274 [gr-qc].

[6] C. L. Wainwright, M. C. Johnson, A. Aguirre, and H. V. Peiris, “Simulating the universe(s) II: phenomenology of cosmic bubble collisions in full General Relativity,” JCAP 1410 (2014) no. 10, 024, arXiv:1407.2950 [hep-th].

[7] M. C. Johnson, H. V. Peiris, and L. Lehner, “Determining the outcome of cosmic bubble collisions in full General Relativity,” Phys. Rev. D85 (2012) 083516, arXiv:1112.4487 [hep-th].

[8] P. Figueras and T. Wiseman, “Stationary holographic plasma quenches and numerical methods for non-Killing horizons,” Phys. Rev. Lett. 110 (2013) 171602, arXiv:1212.4498 [hep-th].

[9] V. Cardoso, L. Gualtieri, C. Herdeiro, and U. Sperhake, “Exploring New Physics Frontiers Through Numerical Relativity,” arXiv:1409.0014 [gr-qc].

[10] L. Lehner and F. Pretorius, “Black Strings, Low Viscosity Fluids, and Violation of Cosmic Censorship,” Phys. Rev. Lett. 105 (2010) 101102, arXiv:1006.5960 [hep-th].

[11] T. Goodale, G. Allen, G. Lanfermann, J. Massó, T. Radke, E. Seidel, and J. Shalf, “The Cactus Framework and Toolkit: Design and Applications,” in Vector and Parallel Processing – VECPAR’2002, 5th International Conference, Lecture Notes in Computer Science. Springer, Berlin, 2003.

[12] J. D. Brown, P. Diener, O. Sarbach, E. Schnetter, and M. Tiglio, “Turduckening black holes: an analytical and computational study,” Phys. Rev. D 79 (2009) 044023, arXiv:0809.3533 [gr-qc].

[13] “Kranc: Kranc Assembles Numerical Code.” http://kranccode.org/.

[14] T. W. Baumgarte and S. L. Shapiro, “On the numerical integration of Einstein’s field equations,” Phys. Rev. D59 (1999) 024007, arXiv:gr-qc/9810065 [gr-qc].

[15] M. Shibata and T. Nakamura, “Evolution of three-dimensional gravitational waves: Harmonic slicing case,” Phys. Rev. D52 (1995) 5428–5444.

[16] B. Giacomazzo and L. Rezzolla, “WhiskyMHD: A New numerical code for general relativistic magnetohydrodynamics,” Class. Quant. Grav. 24 (2007) S235–S258, arXiv:gr-qc/0701109 [gr-qc].

[17] H. P. Pfeiffer, L. E. Kidder, M. A. Scheel, and S. A. Teukolsky, “A Multidomain spectral method for solving elliptic equations,” Comput. Phys. Commun. 152 (2003) 253–273, arXiv:gr-qc/0202096 [gr-qc].
[18] M. Adams, P. Colella, D. T. Graves, J. N. Johnson, N. D. Keen, T. J. Ligocki, D. F. Martin, P. W. McCorquodale, D. Modiano, P. O. Schwartz, T. D. Sternberg, and B. Straalen, “Chombo Software Package for AMR Applications - Design Document, Lawrence Berkeley National Laboratory Technical Report LBNL-6616E.”

[19] D. Alic, C. Bona-Casas, C. Bona, L. Rezzolla, and C. Palenzuela, “Conformal and covariant formulation of the Z4 system with constraint-violation damping,” Phys. Rev. D85 (2012) 064040, arXiv:1106.2254 [gr-qc].

[20] C. Bona, T. Ledvinka, C. Palenzuela, and M. Zacek, “General covariant evolution formalism for numerical relativity,” Phys. Rev. D67 (2003) 104005, arXiv:gr-qc/0302083 [gr-qc].

[21] M. J. Berger and I. Rigoutsos, “An Algorithm for Point Clustering and Grid Generation,” IEEE Trans. Sys. Man & Cyber. 21 (1991) 1278–1286.

[22] M. Babiuc, S. Husa, D. Alic, I. Hinder, C. Lechner, et al., “Implementation of standard testbeds for numerical relativity,” Class. Quant. Grav. 25 (2008) 125012, arXiv:0709.3559 [gr-qc].

[23] M. Alcubierre, “Introduction to 3+1 Numerical Relativity.”

[24] M. Berger and J. Oliger, “Adaptive mesh refinement for hyperbolic partial differential equations,” J. Comput. Phys. (1984) 484–512.

[25] M. J. Berger and P. Colella, “Local adaptive mesh refinement for shock hydrodynamics.”

[26] Y. Zlochower, J. Baker, M. Campanelli, and C. Lousto, “Accurate black hole evolutions by fourth-order numerical relativity,” Phys. Rev. D72 (2005) 024021, arXiv:gr-qc/0505055 [gr-qc].

[27] T. Nakamura, K. Oohara, and Y. Kojima, “General Relativistic Collapse to Black Holes and Gravitational Waves from Black Holes,” Prog. Theor. Phys. Suppl. 90 (1987) 1–218.

[28] C. Bona, J. Masso, E. Seidel, and J. Stela, “A New formalism for numerical relativity,” Phys. Rev. Lett. 75 (1995) 600–603, arXiv:gr-qc/9412071 [gr-qc].

[29] M. Alcubierre, B. Bruegmann, P. Diener, M. Koppitz, D. Pollney, et al., “Gauge conditions for long term numerical black hole evolutions without excision,” Phys. Rev. D67 (2003) 084023, arXiv:gr-qc/0206072 [gr-qc].

[30] S. Bernuzzi and D. Hilditch, “Constraint violation in free evolution schemes: Comparing BSSNOK with a conformal decomposition of Z4,” Phys. Rev. D81 (2010) 084003, arXiv:0912.2920 [gr-qc].
[31] J. D. Brown, “Covariant formulations of BSSN and the standard gauge,” *Phys.Rev. D79* (2009) 104029, arXiv:0902.3652 [gr-qc].

[32] T. W. Baumgarte, P. J. Montero, I. Cordero-Carrion, and E. Muller, “Numerical Relativity in Spherical Polar Coordinates: Evolution Calculations with the BSSN Formulation,” *Phys.Rev. D87* (2013) no.4, 044026, arXiv:1211.6632 [gr-qc].

[33] N. Sanchis-Gual, P. J. Montero, J. A. Font, E. Müller, and T. W. Baumgarte, “Fully covariant and conformal formulation of the Z4 system in a reference-metric approach: comparison with the BSSN formulation in spherical symmetry,” *Phys.Rev. D89* (2014) no.10, 104033, arXiv:1403.3653 [gr-qc].

[34] T. W. Baumgarte and S. L. Shapiro, “Numerical Relativity: Solving Einstein’s Equations on the Computer,”.

[35] J. G. Baker, J. Centrella, D.-I. Choi, M. Koppitz, and J. van Meter, “Gravitational wave extraction from an inspiraling configuration of merging black holes,” *Phys.Rev.Lett. 96* (2006) 111102, arXiv:gr-qc/0511103 [gr-qc].

[36] D. Alic, W. Kastaun, and L. Rezzolla, “Constraint damping of the conformal and covariant formulation of the Z4 system in simulations of binary neutron stars,” *Phys.Rev. D88* (2013) no.6, 064049, arXiv:1307.7391 [gr-qc].

[37] H.-O. Kreiss and J. Oliger, “Comparison of accurate methods for the integration of hyperbolic equations,” *Tellus 24* (1972) no.3, 199–215.

[38] J. Thornburg, “A Fast apparent horizon finder for three-dimensional Cartesian grids in numerical relativity,” *Class.Quant.Grav. 21* (2004) 743–766, arXiv:gr-qc/0306056 [gr-qc].

[39] M. Hannam, S. Husa, F. Ohme, B. Bruegmann, and N. O’Murchadha, “Wormholes and trumpets: The Schwarzschild spacetime for the moving-puncture generation,” *Phys.Rev. D78* (2008) 064020, arXiv:0804.0628 [gr-qc].

[40] S. R. Brandt and E. Seidel, “The Evolution of distorted rotating black holes. 3: Initial data,” *Phys.Rev. D54* (1996) 1403–1416, arXiv:gr-qc/9601010 [gr-qc].

[41] Z. Cao and D. Hilditch, “Numerical stability of the Z4c formulation of general relativity,” *Phys.Rev. D85* (2012) 124032, arXiv:1111.2177 [gr-qc].
A. Plots from code tests

The figures for the code tests described in the paper can be found here.

Figure 7: Robust stability test, BSSN: The right hand figure shows the time evolution of the $L_\infty$ norm of the Hamiltonian constraint, and the left hand figure shows the deviation of $\tilde{\gamma}_{xx}$ from 1. Neither norm grows at an increasing rate with increasing resolution, and so the test is passed.
Figure 8: Robust stability test, CCZ4: Hamiltonian constraint violation and deviation of $\tilde{\gamma}_{xx}$ from 1 for CCZ4 with damping parameter $\kappa_1 = 0.02$. While CCZ4 suppresses the Hamiltonian constraint successfully, the deviation of $\tilde{\gamma}_{xx}$ becomes slightly larger compared to BSSN.

Figure 9: Linear wave test: The left hand figure shows both the analytical solution and the evolved $g_{yy}$ component of the metric at $T = 1000$ at resolution $\rho = 4$, but the two are indistinguishable. The right hand figure shows the absolute value of the error across the grid at $T = 1000$, from which we can see more easily that some small errors in the magnitude and phase have been introduced.
Figure 10: Gauge wave test: The increase in the $L_\infty$ norm of the Hamiltonian constraint means that the BSSN code only remains stable for less than 50 timesteps. Undamped (u) CCZ4, i.e. CCZ4 with $\kappa_1 = 0$, performs similarly. Damped (d) CCZ4 with $\kappa_1 = 1$ is stable for the full 1000 timesteps. The test was performed with initial amplitude of $A = 0.1$, Kreiss-Oliger dissipation coefficient of $\sigma = 0.1$ and a resolution of $\rho = 4$. 

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure10.png}
\caption{Gauge wave test: The increase in the $L_\infty$ norm of the Hamiltonian constraint means that the BSSN code only remains stable for less than 50 timesteps. Undamped (u) CCZ4, i.e. CCZ4 with $\kappa_1 = 0$, performs similarly. Damped (d) CCZ4 with $\kappa_1 = 1$ is stable for the full 1000 timesteps. The test was performed with initial amplitude of $A = 0.1$, Kreiss-Oliger dissipation coefficient of $\sigma = 0.1$ and a resolution of $\rho = 4$.}
\end{figure}
Figure 11: Gowdy wave test, collapsing: The left hand figures show the minimum value of the lapse $\alpha$ across the grid as the spacetime is evolved towards the singularity. As expected, the harmonic gauge causes the evolution to “slow down” as the singularity is approached. The right hand figures show the evolution of the $L_\infty$ norm of the Hamiltonian constraint for two resolutions. The test reaches $T = 1000$ crossing times without crash.
Figure 12: Gowdy wave test, expanding: The left hand figures show the trace of the extrinsic curvature $K$ as the Gowdy wave spacetime is evolved in the collapsing direction. This correctly asymptotes to zero as the spacetime expands, but becomes unstable at around $t = 30$. The right hand figures show the evolution of the $L_{\infty}$ norm of the Hamiltonian constraint for two different resolutions.
Figure 13: Gowdy wave test, convergence: The ratio of the $L_\infty$ norm of the Hamiltonian constraint for the resolutions $\rho = 4$ and $\rho = 2$ is shown, for the expanding and collapsing directions for the BSSN and CCZ4 codes. A value of 16 indicates 4th order convergence, which is demonstrated by the codes initially, although lost at later times by BSSN.

Figure 14: Schwarzschild black hole: The figures show the ADM Mass, Angular momentum and linear momentum (in the $x$ direction) for two different resolutions.
Figure 15: Schwarzschild black hole: The figures show the evolution of the $L_\infty$ norm of the Hamiltonian constraint for the two resolutions along the radial direction. As can be seen, increasing resolution does increase the constraint violation very slightly at the innermost grid, but not significantly, and the violations of the constraints at the grid boundaries are in both cases stable and decrease over time.

Figure 16: Kerr black hole: The figures show the three components of the angular momentum and the mass of the Kerr black hole for two different resolutions.
Figure 17: Kerr black hole: The figures show the evolution of the $L_{\infty}$ norm of the Hamiltonian constraint for the two resolutions take along the radial direction. Again, increasing resolution does increase the constraint violation at the innermost grid, but the violations of the constraints at the grid boundaries are in both cases stable and decrease over time.

Figure 18: Boosted black hole: The figures show the three components of the linear momentum for two different resolutions.
Figure 19: Boosted black hole: The figures show the evolution of the $L_\infty$ norm of the Hamiltonian constraint for the two resolutions, taken along the direction in which the black hole is boosted. It can be seen that the magnitude of the constraint violation is an order of magnitude higher than for the Kerr or Schwarzschild case, because of the frequent regridding required by the movement of the black hole across the grid. Increasing resolution does increase the constraint violation at the innermost grid, but the violations of the constraints at the grid boundaries are in both cases stable and decrease over time.
Figure 20: Choptuik scalar field collapse: The profiles shown for the fields at \( t = 0, 5 \) and 20 differ from those in [23] due to the different gauge conditions used. In the supercritical case we show the snapshot at \( t = 10 \) rather than 20 as this is the point at which the evolution is frozen in the gauge choice in [23]. In the puncture gauge the evolution of the region within the event horizon continues and the result is that the large spike in the field effectively falls into the puncture, resulting in a zero field value at the centre of the coordinate grid.

Figure 21: Choptuik scalar field collapse: The values of the lapse at the centre of the grid are given. It can be seen the the profiles are very similar to those obtained by Alcubierre in [23], and that the one for the supercritical case shows the characteristic collapse of the lapse which is symptomatic of black hole formation.