NRPyElliptic: A Fast Hyperbolic Relaxation Elliptic Solver for Numerical Relativity, I: Conformally Flat, Binary Puncture Initial Data

Thiago Assumpção [1,2,*] Leonardo R. Werneck [3,1,2,†]
Terrence Pierre Jacques [1,2,‡] and Zachariah B. Etienne [3,1,2,§]

[1] *Department of Physics and Astronomy, West Virginia University, Morgantown, WV 26506, USA*
[2] *Center for Gravitational Waves and Cosmology, West Virginia University, Chestnut Ridge Research Building, Morgantown, WV26505, USA*
[3] *Department of Physics, University of Idaho, Moscow, ID 83843, USA*

(Dated: November 5, 2021)

We introduce NRPyElliptic, an elliptic solver for numerical relativity (NR) built within the NRPy+ framework. As its first application, NRPyElliptic sets up conformally flat, binary black hole (BBH) puncture initial data (ID) on a single numerical domain, similar to the widely used TwoPunctures code. Unlike TwoPunctures, NRPyElliptic employs a hyperbolic relaxation scheme, whereby arbitrary elliptic PDEs are trivially transformed into a hyperbolic system of PDEs. As consumers of NR ID generally already possess expertise in solving hyperbolic PDEs, they will generally find NRPyElliptic easier to tweak and extend than other NR elliptic solvers. When evolved forward in (pseudo)time, the hyperbolic system exponentially reaches a steady state that solves the elliptic PDEs. Notably NRPyElliptic accelerates the relaxation waves, which makes it many orders of magnitude faster than the usual constant-wavespeed approach. While it is still ∼12x slower than TwoPunctures at setting up full-3D BBH ID, NRPyElliptic requires only ≈0.3% of the runtime for a full BBH simulation in the Einstein Toolkit. Future work will focus on improving performance and generating other types of ID, such as binary neutron star.

I. INTRODUCTION

To date the LIGO/Virgo gravitational wave (GW) observatories have detected dozens of binary black hole (BBH) mergers [1], and numerical relativity (NR) BBH simulations form a cornerstone of the ensuing data analyses. Such simulations build from formulations of the general relativistic (GR) field equations [2–10] that decompose GR into an initial value problem in 3+1 dimensions. These formulations generally rewrite the GR field equations as a set of time evolution and constraint equations, similar to Maxwell's equations in differential form [11]. Thus, so long as one is provided initial data (ID) that satisfy the Einstein constraints (elliptic PDEs), the evolution equations (hyperbolic PDEs) can propagate them forward in time to construct the spacetime.

Construction of NR ID for realistic astrophysical scenarios is typically a complex and highly specialized task (see [12] for an excellent review). A wide range of approaches have been employed by different groups to set up ID for realistic astrophysical scenarios, including the use of finite difference [13–16], spectral [17–24], and Galerkin [25] methods, which are then combined with special numerical techniques to solve the associated elliptic PDEs. Notably, this generally requires a different skill set than those associated with solving the (hyperbolic) evolution equations, so experts in setting up ID for NR are rarely experts in solving the evolution equations, and *vice versa*.

NRPyElliptic is a new, extensible elliptic solver that sets up initial data for numerical relativity using the same numerical methods employed for solving hyperbolic evolution equations. Specifically, NRPyElliptic implements the hyperbolic relaxation method of [26] to solve complex nonlinear elliptic PDEs for NR ID. The hyperbolic PDEs are evolved forward in (pseudo)time, resulting in an exponential relaxation of the arbitrary initial guess to a steady state that coincides with the solution of the elliptic system. NRPyElliptic solves these equations on highly efficient numerical grids exploiting underlying symmetries in the physical scenario. To this end, NRPyElliptic is built within the SymPy [27]-based NRPy+ code-generation framework [28, 29], which facilitates the solution of hyperbolic PDEs on Cartesian-like, spherical-like, cylindrical-like, or bispherical-like numerical grids. For the purposes of setting up BBH puncture ID, NRPyElliptic makes use of the latter.

Choice of appropriate numerical grids is critically important, as setting up binary compact object ID requires numerically resolving many orders of magnitude in length scale: from the sharp gravitational fields near each compact object, to the nearly flat fields far away. If a constant wavespeed is chosen in the hyperbolic relaxation method (as is typical), the Courant-Friedrichs-Lewy (CFL)-constrained global timestep for the relaxation will be many orders of magnitude smaller than the time required for a relaxation wave to cross the numerical domain. This poses a significant problem as hyperbolic relaxation methods must propagate the relaxation waves across the *entire* numerical domain *several times* to reach convergence. Thus hyperbolic relaxation solvers are gen-

* ta0082@mix.wvu.edu
† leonardo@uidaho.edu
‡ tp0052@mix.wvu.edu
§ zetienne@uidaho.edu

erally far slower than elliptic solvers based on specialized numerical methods.

`NRPyElliptic` solves the hyperbolic relaxation PDEs on a single bispherical-like domain, enabling us to increase the local relaxation wavespeed in proportion to the local grid spacing without violating the CFL condition. As grid spacing in our coordinate system grows *exponentially* away from the two coordinate foci and toward the outer boundary, the relaxation waves accelerate exponentially toward the outer boundary, increasing the solver's overall speed-up over a constant-wavespeed implementation by many orders of magnitude. In fact the resulting performance boost enables `NRPyElliptic` to be useful for setting up high-quality, full-3D BBH puncture ID, though it is still ∼12x slower than the widely used pseudospectral `TwoPunctures` [20] BBH puncture ID solver.

Like `TwoPunctures`, `NRPyElliptic` adopts the conformal transverse-traceless decomposition [12, 30–32] to construct puncture ID for two BHs. In this paper we present both 2D and full 3D validation tests, which demonstrate that `NRPyElliptic` yields identical results to `TwoPunctures` as numerical resolution is increased in both codes.

We also embed `NRPyElliptic` into an `Einstein Toolkit` [33–35] module ("thorn"), called `NRPyEllipticET`, which enables the generated ID to be interpolated onto Cartesian AMR grids within the `Einstein Toolkit`. To demonstrate that `NRPyElliptic` ID are of high fidelity, we first generate 3D BBH puncture ID with both `NRPyEllipticET` and the `TwoPunctures` `Einstein Toolkit` thorns at comparable-accuracy; then evolve the ID forward in time through inspiral, merger, and ringdown using the `Einstein Toolkit` infrastructure; and finally show that the results of these simulations are virtually indistinguishable.

Moving forward, the key advantage to hyperbolic relaxation solvers is their immediate application to solving other elliptic problems. In fact the generality of the hyperbolic relaxation method has already been demonstrated in [26], where it was used to produce ID for many different scenarios of interest, such as scalar fields, Tolman-Oppenheimer-Volkoff (TOV) stars, and binary neutron stars (BNSs). In this work we will focus our discussion on BBH puncture ID, and further evidence of the extensibility of `NRPyElliptic` will be presented in forthcoming papers to generate e.g., BNS ID.

The remainder of this paper is organized as follows. Sec. II introduces the puncture ID formalism, the hyperbolic relaxation method, and our implementation of Sommerfeld (radiation) boundary conditions. In Sec. III we discuss the details of our numerical implementation, including choice of coordinate system and implementation of a grid spacing-dependent wavespeed. We present 2D (axisymmetric) and full 3D validation tests, as well as results from a BBH evolution of our full 3D ID in Sec. IV. We conclude in Sec. V and discuss future work.

II. BASIC EQUATIONS

Throughout this paper we adopt geometrized units, in which $G = c = 1$, and Einstein summation convention such that repeated Latin (Greek) indices imply a sum over all 3 spatial (all 4 spacetime) components.

Consider the 3+1 decomposition of the spacetime metric, with line element

$$ds^2 = -\alpha^2 dt^2 + \gamma_{ij}(dx^i + \beta^i dt)(dx^j + \beta^j dt)\,. \tag{1}$$

Here, α is the lapse function, β^i is the shift vector, and γ_{ij} is the 3-metric.

It is useful to define a conformally related 3-metric $\tilde{\gamma}_{ij}$ via

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

where the scalar function ψ is known as the conformal factor. We adorn geometric quantities associated with $\tilde{\gamma}_{ij}$ with a tilde diacritic. For instance, the Christoffel symbols associated with $\tilde{\gamma}_{ij}$ are computed using

$$\tilde{\Gamma}^k_{\ ij} = \frac{1}{2}\tilde{\gamma}^{lk}(\tilde{\gamma}_{li,j} + \tilde{\gamma}_{lj,i} - \tilde{\gamma}_{ij,l})\,. \tag{3}$$

Likewise, $\tilde{\nabla}_i$ is the associated conformal covariant derivative and $\tilde{R}_{ij}$ the Ricci tensor. All geometric quantities compatible with the physical 3-metric γ_{ij} are written without tildes.

In the limit of vacuum (e.g., BBH) spacetimes, the Hamiltonian and momentum constraint equations can be written as [12]

$$R + K^2 - K_{ij}K^{ij} = 0\,, \tag{4}$$

$$\nabla_j(K^{ij} - \gamma^{ij}K) = 0\,, \tag{5}$$

where K_{ij} is the extrinsic curvature and $K \equiv \gamma^{ij}K_{ij}$ is the mean curvature. Setting up ID for vacuum spacetimes in numerical relativity generally involves solving these constraints, which exist as second-order nonlinear elliptic PDEs.

For the purposes of this paper, we will focus on the puncture ID formalism, in which a set of simplifying assumptions is applied to these constraints, known as the conformal transverse-traceless (CTT) decomposition (see e.g., [12]). For completeness we next apply the CTT approach to Eqs. (4, 5) to derive the constraint equations solved in this paper by `NRPyElliptic`.

A. Puncture Initial Data Formalism

To arrive at the CTT decomposition, we first rewrite the extrinsic curvature as

$$K_{ij} = A_{ij} + \frac{1}{3}\gamma_{ij}K\,, \tag{6}$$

where A_{ij} is the trace-free part of K_{ij}. The conformal counterpart of A_{ij} is defined through the relation

$$A_{ij} \equiv \psi^{-2}\tilde{A}_{ij}\,. \tag{7}$$

The CTT decomposition splits $\tilde{A}_{ij}$ into a symmetric trace-free part $\tilde{M}^{ij}$ and a longitudinal part $(\tilde{\mathbb{L}}V)^{ij}$,

$$\tilde{A}^{ij} = (\tilde{\mathbb{L}}V)^{ij} + \tilde{M}^{ij}\,, \tag{8}$$

where the longitudinal operator $\tilde{\mathbb{L}}$ is defined via

$$(\tilde{\mathbb{L}}V)^{ij} \equiv \tilde{\nabla}^i V^j + \tilde{\nabla}^j V^i \ - \frac{2}{3}\tilde{\gamma}^{ij}\tilde{\nabla}_l V^l\,. \tag{9}$$

Inserting these CTT quantities into the constraint equations (Eqs. 4, 5) yields the generic CTT Hamiltonian and momentum constraint equations

$$\tilde{\nabla}^2\psi - \frac{1}{8}\psi\tilde{R} - \frac{1}{12}\psi^5 K^2 + \frac{1}{8}\psi^{-7}\tilde{A}_{ij}\tilde{A}^{ij} = 0\,, \tag{10}$$

$$\tilde{\Delta}_{\mathbb{L}}V^i - \frac{2}{3}\psi^6\tilde{\nabla}^i K + \tilde{\nabla}_j\tilde{M}^{ij} = 0\,, \tag{11}$$

where $\tilde{R}$ is the conformal Ricci scalar and the operator $\tilde{\Delta}_{\mathbb{L}}$ is defined as

$$\tilde{\Delta}_{\mathbb{L}}V^i \equiv \tilde{\nabla}_j(\tilde{\mathbb{L}}V)^{ij} = \tilde{\nabla}^2 V^i + \frac{1}{3}\tilde{\nabla}^i(\tilde{\nabla}_j V^j) + \tilde{R}^i{}_j V^j\,. \tag{12}$$

The degrees of freedom in this formulation include choice of $\tilde{M}^{ij}$, K, and $\tilde{\gamma}_{ij}$. Here we consider puncture ID, which assume maximal slicing ($K = 0$), asymptotic flatness ($\psi|_{r\to\infty} = 1$), and conformal flatness

$$\tilde{\gamma}_{ij} = \hat{\gamma}_{ij}\,, \tag{13}$$

where $\hat{\gamma}_{ij}$ is the flat spatial metric. In addition the assumption $\tilde{M}^{ij} = 0$ is made, yielding Hamiltonian and momentum constraint equations of the form

$$\hat{\nabla}^2\psi + \frac{1}{8}\psi^{-7}\tilde{A}_{ij}\tilde{A}^{ij} = 0\,, \tag{14}$$

$$\hat{\nabla}^2 V^i + \frac{1}{3}\hat{\nabla}^i(\hat{\nabla}_j V^j) = 0\,, \tag{15}$$

where $\hat{\nabla}_i$ is the covariant derivative compatible with $\hat{\gamma}_{ij}$. Bowen and York [36] showed that the momentum constraint is solved for a set of N_p punctures with a closed-form expression for the extrinsic curvature. This expression can be written in terms of $\vec{V}$ as follows

$$\vec{V} = \sum_{n=1}^{N_p}\left(-\frac{7}{4|\vec{x}_n|}\vec{P}_n - \frac{\vec{x}_n\cdot\vec{P}_n}{4|\vec{x}_n|^3}\vec{x}_n + \frac{1}{|\vec{x}_n|^3}\vec{x}_n\times\vec{S}_n\right), \tag{16}$$

where $\vec{x}_n = (x_n - x, y_n - y, z_n - z)$, $\vec{P}_n$, and $\vec{S}_n$ are the displacement relative to the origin (i.e., $(x, y, z) = (0, 0, 0)$), linear momentum, and spin angular momentum of puncture n, respectively.

The Hamiltonian constraint equation (Eq. 14) must be solved numerically, but ψ becoming singular at the location of each puncture could spoil the numerical solution. Early attempts excised the singular terms from the computational domain (see [12] and references therein), but modern approaches generally follow [37] in splitting the conformal factor into a singular and a non-singular piece,

$$\psi = \psi_{\rm singular} + u \equiv 1 + \sum_{n=1}^{N_p}\frac{m_n}{2|\vec{x}_n|} + u\,, \tag{17}$$

where m_n is the bare mass of the $n^{\rm th}$ puncture. The Hamiltonian constraint equation, which can then be solved for the non-singular part u, reads

$$\hat{\nabla}^2 u + \frac{1}{8}\tilde{A}_{ij}\tilde{A}^{ij}(\psi_{\rm singular} + u)^{-7} = 0\,. \tag{18}$$

B. Hyperbolic relaxation method

We now describe the basic hyperbolic relaxation method of [26]. Consider the system of elliptic equations

$$\mathcal{L}_E\vec{u} - \vec{\rho} = 0\,, \tag{19}$$

where $\mathcal{L}_E$ is an elliptic operator, $\vec{u}$ is the vector of unknowns, and $\vec{\rho}$ is the vector of source terms. The hyperbolic relaxation method replaces Eq. (19) with the hyperbolic system of equations

$$\partial_t^2\vec{u} + \eta\partial_t\vec{u} = c^2\left(\mathcal{L}_E\vec{u} - \vec{\rho}\right), \tag{20}$$

where η is an exponential damping parameter (with units of $1/t$ [38]) and c is the wavespeed. The variable t behaves as a time variable in this hyperbolic system of equations and is referred to as a *relaxation* (as opposed to physical) time. As noted in [26], the damping parameter η that maximizes dissipation is dictated by the length scale of the grid domain and is easily determined when the wavespeed is constant. Through numerical experimentation, we found that the choice $\eta = 12.5/M$ minimizes the required relaxation time when using the spatially-dependent wavespeed prescription introduced in Sec. III B and detailed in Appendix B.

If appropriate boundary conditions are chosen, when Eq. (20) is evolved forward in (pseudo)time, the damping ensures that a steady state is eventually reached exponentially fast such that $\partial_t u \to 0$ and $\partial_t^2 u \to 0$. Thus u relaxes to a solution to the original elliptic problem. To this end, we adopt Sommerfeld (outgoing radiation) boundary conditions (BCs) for spatial boundaries, as described in Sec. II C; what remains is a choice of initial conditions. As this is a relaxation method, any smooth choice should suffice. For simplicity, in this work we set trivial initial conditions $\vec{u} = \partial_t\vec{u} = \vec{0}$.

To complete our expression of these equations in preparation for a full numerical implementation, we rewrite Eq. (20) as a set of two first-order (in time) PDEs

$$\begin{aligned}\partial_t\vec{u} &= \vec{v} - \eta\vec{u}\,,\\ \partial_t\vec{v} &= c^2\left(\mathcal{L}_E\vec{u} - \vec{\rho}\right),\end{aligned} \tag{21}$$

so that the method of lines (Sec. III) can be immediately used to propagate the solution forward in (pseudo)time

until a convergence criterion has been triggered (indicating numerical errors associated with the solution to the elliptic equation are satisfactorily small).

As a simple example, consider Poisson's equation, for which $\mathcal{L}_E\vec{u} = \mathcal{L}_E u = \nabla^2 u = u^{,i}_{\ ,i}$. This PDE can be easily made covariant ("comma goes to semicolon rule"):

$$\hat{\nabla}^2 u = u^{\ ;i}_{;i} = \rho\,, \tag{22}$$

where $\hat{\nabla}_i$ is the covariant derivative compatible with $\hat{\gamma}_{ij}$. In this way, the Laplace operator is expanded as

$$\hat{\nabla}^i\hat{\nabla}_i u = \hat{\gamma}^{ij}\hat{\nabla}_i\hat{\nabla}_j u = \hat{\gamma}^{ij}\left(\partial_i\partial_j u - \hat{\Gamma}^k_{\ ij}\partial_k u\right), \tag{23}$$

with $\hat{\Gamma}^k_{\ ij}$ the Christoffel symbols associated with $\hat{\gamma}_{ij}$. Poisson's equation is then written as the system

$$\boxed{\begin{aligned}\partial_t u &= v - \eta u\\ \partial_t v &= c^2\left(\hat{\nabla}^2 u - \rho\right)\end{aligned}}\,. \tag{24}$$

Writing the PDEs covariantly enables the hyperbolic relaxation method to be applied in coordinate systems that properly exploit near-symmetries. For this purpose we adopt a reference metric $\hat{\gamma}_{ij}$, which is chosen to be the flat spatial metric in the given coordinate system we are using. In this way, single compact object ID can be solved in spherical or cylindrical coordinates (using spherical or cylindrical reference metrics respectively), and binary ID can be solved in bispherical-like coordinates.

Truly the power of the hyperbolic relaxation method is its easy and immediate extension to complex, nonlinear elliptic PDEs. Case in point: as derived in Sec. II A, ID for two punctures are constructed by solving the Hamiltonian constraint

$$\hat{\nabla}^2 u + \frac{1}{8}\tilde{A}_{ij}\tilde{A}^{ij}(\psi_{\rm singular} + u)^{-7} = 0\,, \tag{25}$$

for u. This elliptic PDE is nonlinear, but is trivially embedded within the hyperbolic relaxation prescription via[1]

$$\boxed{\begin{aligned}\partial_t u &= v - \eta u\\ \partial_t v &= c^2\left[\tilde{\nabla}^2 u + \frac{1}{8}\tilde{A}_{ij}\tilde{A}^{ij}\left(\psi_{\rm singular} + u\right)^{-7}\right]\end{aligned}}\,. \tag{26}$$

Note that just like in the case of Poisson's equation, $\tilde{\nabla}^2 u$ is expanded as in Eq. (23).

[1] Recall in the previous section we adopted the tilde instead of the hat diacritic to denote the flat metric, consistent with the general convention in the literature. Both can be used interchangeably here, as $\tilde{\gamma}_{ij} = \hat{\gamma}_{ij}$.

C. Boundary conditions

Similar to both the hyperbolic relaxation method implemented in [26] and the `Einstein Toolkit` BC driver `NewRad` [33, 34, 39], spatial BCs are applied to the time derivatives of the evolved fields instead of the fields directly. Consequently the desired BC is only satisfied by the steady state solution.

For example, assume that at $\partial\Omega$, the boundary of our numerical domain, we wish to impose Dirichlet BCs of the form

$$\begin{aligned}\vec{u}\big|_{\partial\Omega} &= \vec{a}\,,\\ \vec{v}\big|_{\partial\Omega} &= \vec{b}\,,\end{aligned} \tag{27}$$

for some constant vectors $\vec{a}$ and $\vec{b}$. In our implementation, these would be imposed as

$$\begin{aligned}\partial_t\vec{u}\big|_{\partial\Omega} &= \vec{u} - \vec{a}\,,\\ \partial_t\vec{v}\big|_{\partial\Omega} &= \vec{v} - \vec{b}\,.\end{aligned} \tag{28}$$

Upon reaching the steady state, $\partial_t u\big|_{\partial\Omega} = 0 = \partial_t v\big|_{\partial\Omega}$, and we recover the desired BCs.

When applying the hyperbolic relaxation method to solve the Einstein constraint equations, outgoing radiation BCs are most appropriate, as they allow the outgoing relaxation wave fronts to pass through the boundaries of the numerical domain with minimal reflection.

Radiation (Sommerfeld) BCs generally assume that near the boundary each field f behaves as an outgoing spherical wave, and our implementation follows the implementation within `NewRad`, building upon the *ansatz*:

$$f = f_0 + \frac{w(r-ct)}{r} + \frac{K}{r^2}\,, \tag{29}$$

where $f_0 = \lim_{r\to\infty} f$, $w(r-ct)/r$ satisfies the spherical wave equation for an outgoing spherical wave, and K/r^2 models higher-order radial corrections.

Just as in the case of Dirichlet BCs, we apply Sommerfeld BCs to the time derivative of the fields. Appendix A walks through the full derivation for applying Sommerfeld BCs to any field $\partial_t f$, as well as its numerical implementation. Based on Eq. (A10), Sommerfeld BCs for a generic hyperbolic relaxation of solution vector $\vec{u}$ takes the form

$$\begin{aligned}\partial_t\vec{u}\big|_{\partial\Omega} &= -\frac{c}{r}\left[r\partial_r\vec{u} + (\vec{u} - \vec{u}_0)\right] + \frac{\vec{k}_u}{r^3}\,,\\ \partial_t\vec{v}\big|_{\partial\Omega} &= -\frac{c}{r}\left[r\partial_r\vec{v} + (\vec{v} - \vec{v}_0)\right] + \frac{\vec{k}_v}{r^3}\,,\end{aligned} \tag{30}$$

where $\vec{k}_u$ and $\vec{k}_v$ are constant vectors computed at each boundary point for each field within the $\vec{u}$ and $\vec{v}$ vectors using Eq. (A15).

III. NUMERICAL IMPLEMENTATION

`NRPyElliptic` exists as both a standalone code and an `Einstein Toolkit` module ("thorn"), `NRPyEllipticET`. `NRPyEllipticET` incorporates the standalone code into the `Einstein Toolkit`, solving the elliptic PDE entirely within `NRPyElliptic`'s `NRPy+`-based infrastructure. Once the solution has been found, `NRPyEllipticET` uses the `Einstein Toolkit`'s built-in ($3^{\rm rd}$-order Hermite) interpolation infrastructure to interpolate the solution from its native, bispherical-like grids to the Cartesian AMR grids used by the `Einstein Toolkit`. From there, the data can be evolved forward in time using any of the various BSSN or CCZ4 `Einstein Toolkit` thorns. Both standalone and `Einstein Toolkit` thorn versions of the `NRPyElliptic` code are fully documented in pedagogical `Jupyter` notebooks. Henceforth we will describe our implementation of the standalone version.

Our implementation of Eqs. (26) within `NRPyElliptic` leverages the `NRPy+` framework [28, 29] to convert these expressions, written symbolically using `NRPy+`'s Einstein-like notation, into highly optimized C-code kernels (`SymPy` [27] serves as `NRPy+`'s computer algebra system backend). Notably `NRPy+` supports the generation of such kernels with single instruction, multiple data (SIMD) intrinsics and common sub-expression elimination (CSE). Further `NRPy+` supports arbitrary-order finite-difference kernel generation, and we use $10^{\rm th}$-order to approximate all spatial derivatives in this work. The time evolution is performed using the method of lines (MoL) infrastructure within `NRPy+`, choosing its fourth-order (explicit) Runge-Kutta implementation (RK4).

`NRPy+` supports a plethora of different reference metrics, enabling us to solve our covariant hyperbolic PDEs (Eqs. 26) in a large variety of Cartesian-like, spherical-like, cylindrical-like, or bispherical-like coordinate systems. This in turn enables the user to fully take advantage of symmetries or near-symmetries of any given problem. For example, for problems involving near-spherical symmetry we have used spherical-like coordinates (e.g., log-radial spherical coordinates). In this work, we make use of the prolate spheroidal-like (i.e., "bispherical-like") coordinate system in `NRPy+` called `SinhSymTP`, described in detail in Sec. III A. This allows us to solve the elliptic problem for two puncture black hole initial data within a single domain, similar to the `TwoPunctures` code.

Note that the wavespeed c appearing e.g., in Eqs. (26), need not be constant. In curvilinear coordinates where the grid spacing is not constant, the CFL stability criterion remains satisfied if the wavespeed is adjusted in proportion to the local grid spacing. As the grid spacing in the `SinhSymTP` coordinates adopted here grows *exponentially* with distance from the strong-field region, the wavespeed grows exponentially as well. As a result, relaxation waves accelerate exponentially to the outer boundary, significantly speeding up the convergence to the solution of the elliptic PDE. Our implementation of this technique is detailed in Sec. III B.

A. Coordinate system

Like `TwoPunctures`, `NRPyElliptic` adopts a modified version of prolate spheroidal (PS) coordinates when setting up two-puncture ID. However, these coordinate systems are distinct both from each other and from PS coordinates. Here we elucidate the differences and similarities.

Consider first PS coordinates (μ,ν,φ), which are related to Cartesian coordinates (x,y,z) via [40]

$$\begin{aligned} x &= a\sinh\mu\sin\nu\cos\varphi\,,\\ y &= a\sinh\mu\sin\nu\sin\varphi\,,\\ z &= \left(a^2\sinh^2\mu + a^2\right)^{1/2}\cos\nu\,. \end{aligned} \tag{31}$$

Here, $\mu\in[0,\infty)$, $\nu\in[0,\pi)$, $\varphi\in[0,2\pi)$, and the two foci of the coordinate system are located at $z=\pm a$.

`TwoPunctures` [20] adopts a PS-like coordinate system, which is written in terms of coordinate variables $A\in[0,1)$, $B\in[-1,1]$, and $\varphi\in[0,2\pi)$. `TwoPunctures` coordinates are related to Cartesian via[2]

$$\begin{aligned} x &= b\frac{2A}{1-A^2}\frac{1-B^2}{1+B^2}\sin\varphi\,,\\ y &= b\frac{2A}{1-A^2}\frac{1-B^2}{1+B^2}\cos\varphi\,,\\ z &= b\frac{A^2+1}{A^2-1}\frac{2B}{1+B^2}\,. \end{aligned} \tag{32}$$

The two foci of `TwoPunctures` coordinates are situated at $z=\pm b$. Of note, the coordinate A is compactified with $|x|,|y|,|z|\to\infty$ as $A\to 1$. Similar to the term $\sin\nu$ (where $\nu\in[0,\pi)$) in PS coordinates, the $(1-B^2)/(1+B^2)$ (where $B\in[-1,1]$) term is a concave down curve with a maximum of 1 at the midpoint of the range of B and zeroes at the endpoints $B=\pm1$. Unlike PS coordinates, however, the coordinate system is not periodic in the variable B.

`NRPyElliptic` adopts the `NRPy+` PS-like coordinate system `SinhSymTP` $(\mathtt{x_1},\mathtt{x_2},\mathtt{x_3})$ with $\mathtt{x_1}\in[0,1]$, $\mathtt{x_2}\in[0,\pi]$, and $\mathtt{x_3}\in[-\pi,\pi]$. These are related to PS coordinates via

$$\begin{aligned} a\sinh\mu = \tilde{r} &\equiv \mathcal{A}\frac{\sinh(\mathtt{x_1}/w)}{\sinh(1/w)}\,,\\ \nu &= \mathtt{x_2}\,,\\ \varphi &= \mathtt{x_3}\,, \end{aligned} \tag{33}$$

with $\tilde{r}\in[0,\mathcal{A}]$. Introducing the parameter b, we obtain

$$\begin{aligned} x &= \tilde{r}\sin(\mathtt{x_2})\cos(\mathtt{x_3})\,,\\ y &= \tilde{r}\sin(\mathtt{x_2})\sin(\mathtt{x_3})\,,\\ z &= \left(\tilde{r}^2+b^2\right)^{1/2}\cos(\mathtt{x_2})\,. \end{aligned} \tag{34}$$

[2] We swap the x and z coordinates of [20] to simplify comparison.

Thus the foci exist at $z = \pm b$ and are decoupled from the grid scaling. As such, `SinhSymTP` allows one to decrease or increase the value of the parameter w to increase or decrease the grid point sampling near the foci, respectively. Finally the $\mathcal{A}$ parameter specifies the domain size.

As illustrated in Fig. 1, like PS and `TwoPunctures` coordinates, `SinhSymTP` coordinates become spherical in the region far from the foci. Note also that regular spherical coordinates (with a non-uniform radial coordinate) are fully recovered by setting $b = 0$. As with

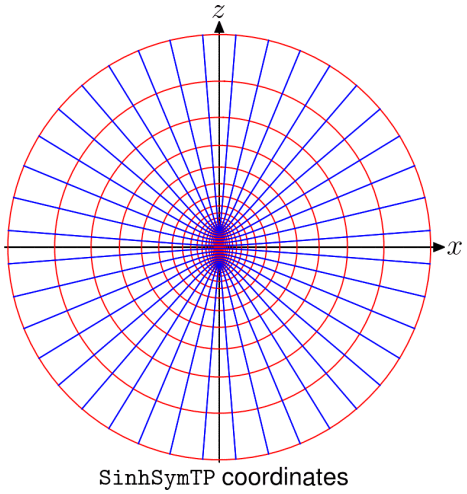

FIG. 1. Curves of constant $\mathtt{x}_1$ (red) and constant $\mathtt{x}_2$ (blue) using a cell-centered grid structure with $\mathtt{x}_3 = 0$.

`TwoPunctures`, `NRPyEllipticET` possesses the option to rotate the coordinate system, to situate the punctures on either the x-axis ($x = \pm b$) or the z-axis ($z = \pm b$).

For all cases considered here, the outer boundary is set to 10^6 (i.e., $\mathcal{A} = 10^6$), and the grid point spacing parameter w is set to 0.07. We set b so that the foci match the punctures' positions and, when interpolating the ID to the `Einstein Toolkit` grids, we adjust the origin of the coordinate system to coincide with the center of mass of the punctures, as is conventional.

B. Wavespeed

To propagate the hyperbolic system of equations forward in (pseudo)time from the chosen initial conditions, we make use of `NRPy+`'s method of lines implementation. Specifically we choose the explicit fourth-order Runge-Kutta (RK4) method. As this is an explicit method, and we use three dimensions in space, the steps in time Δt are constrained by the CFL inequality:

$$\frac{c\Delta t}{\Delta s_{\rm min}} \leq \mathcal{C}_0 \,, \tag{35}$$

where c is the local wavespeed, and $\mathcal{C}_0$ is the CFL factor, which depends on the explicit time stepping method and the dimensionality of the problem. Empirically we find that $\mathcal{C}_0 = 0.7$ ensures both stability and large time steps for both 2D and 3D cases presented in this paper, with one exception: when the 3D case is pushed to very high resolution. In the single highest-resolution 3D case in this work, we find $\mathcal{C}_0$ must be lowered to 0.55 for stability.

Further, $\Delta s_{\rm min}$ is the minimum proper distance between neighboring points in our curvilinear coordinate system:

$$\Delta s_{\rm min} = \min\left(h_1 \Delta \mathtt{x}_1,\, h_2 \Delta \mathtt{x}_2,\, h_3 \Delta \mathtt{x}_3\right) \,, \tag{36}$$

where h_i and $\Delta \mathtt{x}_\mathtt{i}$ are the i-th scale factor and grid spacing of the flat space metric, respectively.

The global relaxation time step $\Delta t_{\rm glob}$ is given by the minimum value of $\mathcal{C}_0 \Delta s_{\rm min}/c$ on our numerical grid. As we adopt prolate spheroidal-like coordinates, the global $\Delta s_{\rm min}$ occurs precisely at the foci of the coordinate system. At this point, for simplicity we set the wavespeed $c = 1$. As this is merely a relaxation (as opposed to a physical) wavespeed, we increase c in proportion to the local $\Delta s_{\rm min}$, which grows exponentially away from the foci. In this way we maintain satisfaction of the CFL inequality while greatly improving the performance of the relaxation method. Details and implications of our implementation are described in Appendix B.

IV. RESULTS

Validation of `NRPyElliptic` is performed in two stages. First we generate initial data (ID) for a given physical scenario with the widely used `TwoPunctures` [20] code, increasing resolution on the `TwoPunctures` grids until roundoff error dominates its numerical solution of Eq. (18), u. We refer to this high-resolution result as the trusted solution. Second we generate the same ID with `NRPyElliptic`, and demonstrate that its results approach the trusted solution at the expected convergence rate.

We repeat this procedure twice: first for an axisymmetric case of two equal-mass BHs with spin vectors collinear with their separation vector, and second for a full 3D case involving a GW150914-like unequal-mass, quasi-circular, spinning BBH system. To demonstrate the fidelity of the latter case, we first generate `NRPyElliptic` and `TwoPunctures` ID at similar levels of accuracy. Then, using the `Einstein Toolkit` [33–35] we evolve the ID through inspiral, merger, and ringdown, and compare the results. Finally we note that $M = M_+ + M_-$ is defined as the sum of individual ADM masses of the punctures (Eq. (83) of Ref. [20]).

A. Axisymmetric initial data

In this test, we generate initial data (ID) for a scenario symmetric about the z-axis: two equal-mass punctures at

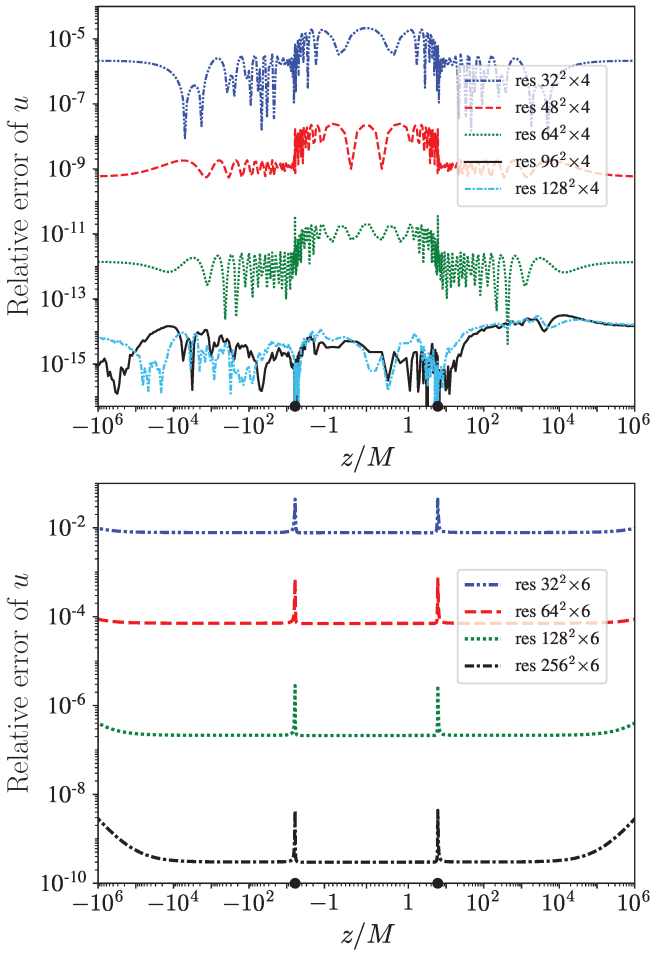

FIG. 2. Calibration of `TwoPunctures` and `NRPyElliptic` solutions, axisymmetric ID study. **Top**: Relative errors between a super-high resolution ($196^2\times4$) and lower-resolution results from the `TwoPunctures` code. **Bottom**: Calibration of `NRPyElliptic` solutions at various resolutions ($\mathtt{N}_1\times\mathtt{N}_2\times6$), comparing against the trusted solution (i.e., the `TwoPunctures` result at $96^2\times4$ resolution). The horizontal axis is logarithmic in the range $|z|>1$ and linear otherwise. Further, the black dots on the horizontal axes denote the puncture BH positions (also the locations of the coordinate foci). `TwoPunctures` data were rotated so that the punctures and the foci lie on the z axis.

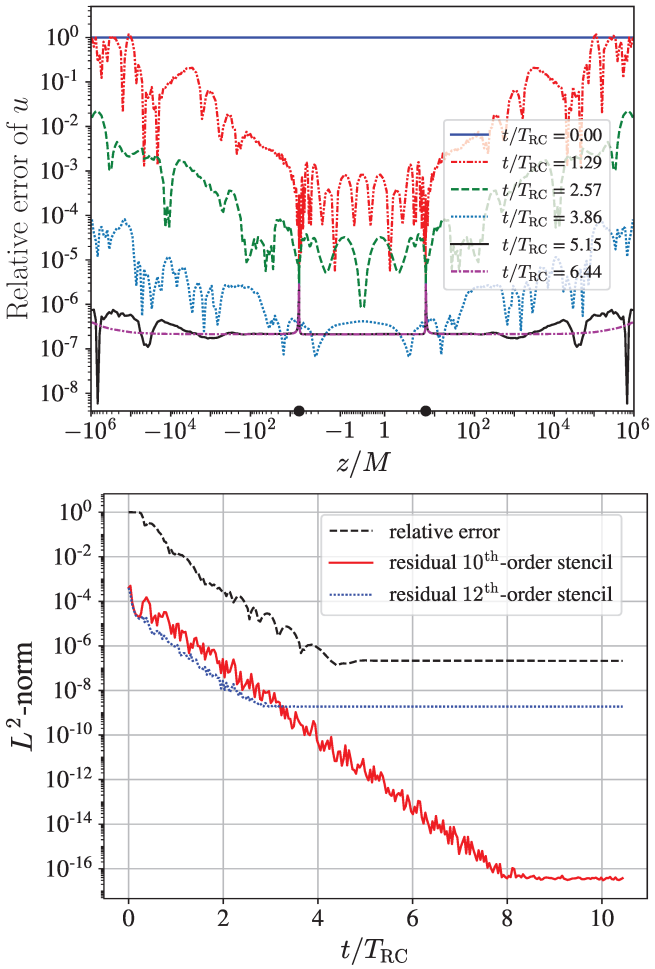

FIG. 3. Axisymmetric ID study. **Top**: Snapshots of the relative error between the `NRPyElliptic` solution at $128^2\times4$ resolution and the trusted solution at different times. `TwoPunctures` data were rotated so that the punctures and the foci lie on the z axis. **Bottom**: Black dashed curve: L^2-norm of the same relative error as the top panel, computed within a sphere with radius $r=100M$ and as a function of time. Blue (dotted) and red (solid) curves: L^2-norms of residuals (Eq. 18) computed within the same sphere at two different finite-difference orders as a function of time.

rest, with spin components $\vec{S}_\pm=\pm0.4M_\pm^2\hat{z}$ and initial positions $z_\pm=\pm6M$.[3]

We start by constructing the trusted solution u, against which we will compare results from `NRPyElliptic`. The trusted solution is generated by increasing the resolution of the `TwoPunctures` numerical grids until roundoff errors dominate. As we adopt double-precision arithmetic, this occurs when relative errors reach levels of roughly 10^{-14}. The top panel of Fig. 2 indicates that this occurs at a `TwoPunctures` grid resolution of $N_A\times N_B\times N_\phi=96^2\times4$.[4]

After obtaining this $96^2\times4$ trusted solution, we next compare it against `NRPyElliptic` at various resolutions in the bottom panel of the figure. Notice that the numerical errors drop by roughly 2^9 each time the resolution is doubled, indicating that numerical convergence is dom-

[3] The bare mass of each puncture is $m=0.456428$ and the total ADM mass is $M_{\rm ADM}=0.979989M$.

[4] To ensure we reach the level where roundoff errors dominate, the maximum number of iterations of the Newton-Raphson method (`Newton_maxit` parameter) is set to 10, and the tolerance (`Newton_tol` parameter) is to $\times10^{-16}$, for the axisymmetric case. Also we fix $N_\phi=4$, following an axisymmetric example in Ref. [20].

inated by our choice of 10^{th}-order finite-difference (FD) stencils. That perfect 10^{th}-order convergence is not obtained is unsurprising, as our outer boundary condition is approximate and implements its own 6^{th}-order FD stencils.[5]

The exponential convergence in time of the hyperbolic relaxation method can be readily seen in Fig. 3. In the top panel snapshots of the relative error at resolution $128^2\times6$ are shown as a function of relaxation-wave-crossing times, or RCTs (i.e., the time required for a relaxation wave to cross the numerical domain; see Appendix B). During the first RCT, the errors decrease (increase) in the region near (away from) the punctures and subsequently drop exponentially in time. Numerical errors are consistently smaller in the region near the origin due to the extreme grid focusing there. Notice that after $\sim$4 RCTs the relative errors between the punctures have already reached the same order of magnitude as that of the fully relaxed solution, and the remaining relaxation time (40% of the total) is spent decreasing the relative errors away from the punctures.

Coincident with the exponential relaxation along the z-axis, the L^2-norm of both the relative errors and residual (left-hand side of Eq. 25), computed inside a sphere of radius $100M$ around the origin (situated at the center of mass), displays a steady exponential drop, as illustrated in the bottom panel of Fig. 3. We find that when the FD order for computing the residual matches the one used by the relaxation algorithm, the L^2-norm of the residual continues to drop even after the steady state has been reached, until roundoff errors dominate this measure of residual. This indicates that the residual has become overspecialized to the approximation adopted for the FD operators. To ameliorate this, we also compute the residual using 12^{th}-order stencils, finding this measure of the residual to have more consistent behavior with the true error (i.e., the relative error against the trusted solution). We repeated this analysis at 8^{th} and 14^{th} finite-differencing order and found the same qualitative behavior as 12^{th} order.

B. Full 3D initial data

The second test consists of two punctures with mass ratio $q = 36/29$, in a quasicircular orbit emulating the GW150914 event [41]. The punctures are located at $z = \pm5M$, with spin components $\vec{S}_+ = 0.31M_+^2\hat{y}$ and $\vec{S}_- = -0.46M_-^2\hat{y}$. [6]

The validation procedure for both `TwoPunctures` and `NRPyElliptic` follows the prescription used in the axisymmetric case. Since the full-3D system does not posses axial symmetry, the number of grid points along the azimuthal direction is no longer minimal. As indicated in the top panel of Fig. 4, the trusted `TwoPunctures` solution has resolution $96^2\times16$, and we found that using $N_\phi > 16$ does not lower the relative errors further. In `NRPyElliptic` (bottom panel of Fig. 4), the optimal resolution in the azimuthal direction was also found to be $\texttt{N}_3 = 16$, except for when the highest resolution ($256^2 \times 24$) was chosen. We find this solution to have smaller relative errors when compared to the $256^2 \times 16$ resolution data (not shown), and requires a lower CFL-factor of $\mathcal{C}_0 = 0.55$ for numerical stability. With truncation errors dominated by sampling in the $\texttt{x}_1$ and $\texttt{x}_2$ directions, we find roughly 9^{th}-order convergence for this case as well.

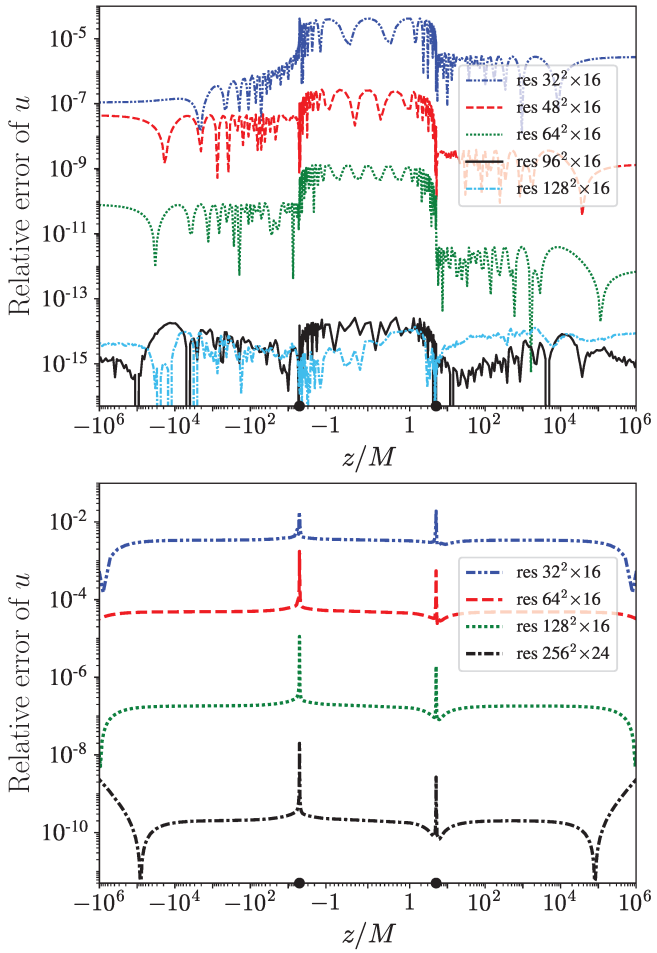

FIG. 4. Same as Fig. 2, but for GW150914-like, full-3D BBH initial data.

[5] `NRPy+` currently requires the minimum number of grid points in any given direction to be even and larger than `FD_order`/2, where `FD_order` order of the finite difference scheme. Thus we set $\texttt{N}_3 = 6$ despite the system being axisymmetric. This requirement may be relaxed in the future.

[6] The bare (local ADM) masses of the punctures are $m_+ = 0.518419$ ($M_+ = 0.553846$) and $m_- = 0.391936$. ($M_- = 0.446154$). The total ADM mass is $M_{\text{ADM}} = 0.989946M$. To elicit a quasicircular orbit, linear momenta are set to $\vec{P}_\pm = \pm P_\phi\hat{x} \pm P_r\hat{z}$, where $P_\phi = 9.53\times10^{-2}$ and $P_r = -8.45\times10^{-4}$, in code units.

As expected, the relaxation to the steady state has exponential convergence in (pseudo)time as depicted in both panels of Fig. 5. Although the relaxation requires a larger number of RCTs in the full-3D case, its qualitative behavior is the same.

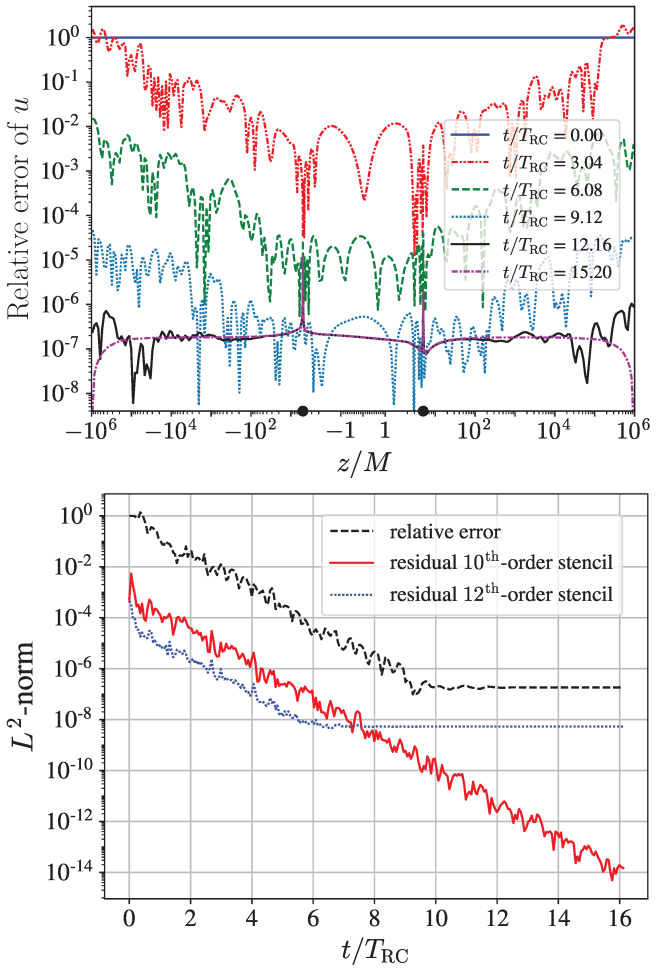

FIG. 5. Same as Fig. 3, but for GW150914-like, full-3D BBH initial data, with `NRPyElliptic` resolution of $128^2 \times 16$.

Next, to demonstrate the fidelity of full-3D initial data generated by `NRPyElliptic`, we first generate ID for the GW150914-like BBH scenario using both `NRPyElliptic` and `TwoPunctures`. These ID are chosen to be of comparable quality, as shown in Fig. 6. Precise numerical parameters for the ID exactly follow the GW150914 `Einstein Toolkit` gallery example [20, 33, 42–50], for which complete results are freely available online.[7] Notably the `TwoPunctures` code required slightly less than a minute to generate the ID (we input the bare masses directly), while `NRPyElliptic` required about 10 minutes, for a difference in performance of ≈12x.[8]

After generation, the ID from both codes are interpolated onto the evolution grids, which consist of a set of `Carpet` [44]-managed Cartesian AMR patches in the strong field region, with resolution around the punctures of $\approx M/52$. These AMR patches are surrounded by a `Llama` [45]-managed cubed spheres grid in the weak-field region. When interpolating the ID onto these grids, `TwoPunctures` adopts a high-accuracy spectral interpolator (see e.g., [51]), and `NRPyElliptic` uses the third-order Hermite polynomial interpolator from the `AEILocalInterp` thorn. As our choice of Hermite interpolation order results in far less accuracy, it introduces error to the Hamiltonian constraint violation at time $t = 0$ on the orbital plane, as shown in the left panel of Fig. 7.

We then evolve these ID forward in time using the `McLachlan` BSSN thorn [48]. Notice that after the first orbital period the Hamiltonian constraint violations on the orbital plane are essentially identical (right panel of Fig. 7), indicating that numerical errors associated with the evolution quickly dominate ID errors.

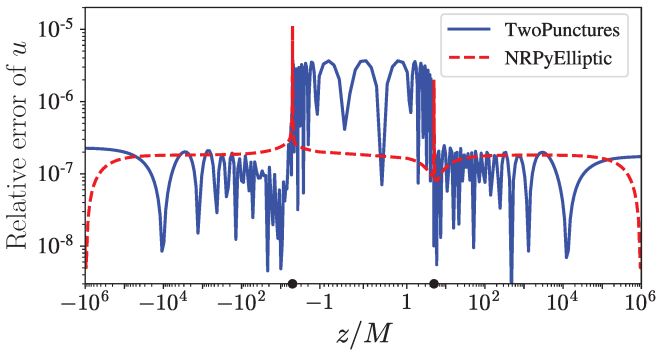

FIG. 6. Initial data quality of the GW150914-like BBH 3D scenario evolved with the `Einstein Toolkit`, showing the relative error of u against the trusted `TwoPunctures` solution. Here `TwoPunctures` and `NRPyElliptic` ID used resolutions of $38^2 \times 16$ and $128^2 \times 16$ respectively. Note that `TwoPunctures` data were rotated so that the punctures are positioned on the z axis.

Finally, to demonstrate that `NRPyElliptic`'s 3D data are of sufficiently high fidelity, we demonstrate that when they are evolved forward in time the results are indistinguishable from an evolution of `TwoPunctures` ID at comparable initial accuracy. For instance in Fig. 8 we show the trajectory of the less-massive BH and the dominant mode ($\ell = 2$, $m = 2$) of the Weyl scalar ψ_4, for both evolutions. We find that quantities extracted from the evolution of `NRPyElliptic` ID to be in excellent agreement with results from evolution of ID from the trusted `TwoPunctures` code. Although not shown, the trajectory of the more-massive BH for both simulations are

[7] See https://einsteintoolkit.org/gallery/bbh/index.html for more details.

[8] Benchmarks were performed on a 16-core AMD Ryzen 9 3950X CPU; the speed-up factor was found to be very similar on other CPUs as well.

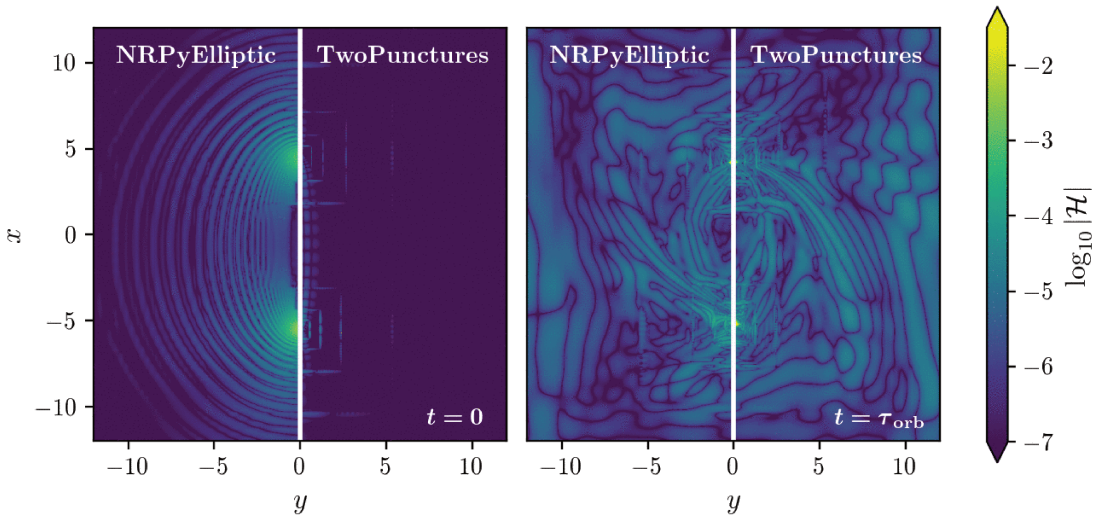

FIG. 7. Hamiltonian constraint violation $\log_{10}|\mathcal{H}|$ in orbital plane after interpolation to AMR grids at $t = 0$ (**left**) and after one orbit (**right**). Note that `NRPyElliptic` data were rotated so that the punctures are positioned on the x axis.

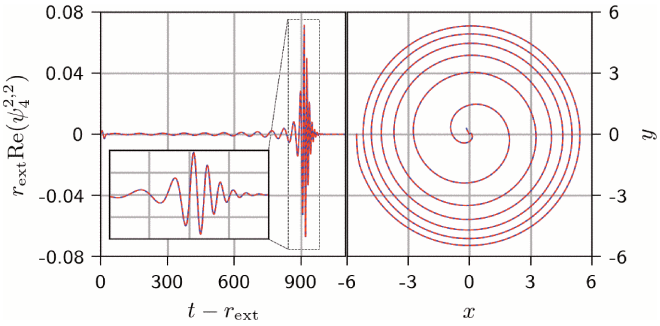

FIG. 8. Simulation results with ID by `TwoPunctures` (solid, blue) and `NRPyElliptic` (dashed, orange-red). **Left**: Dominant mode ($\ell = 2$, $m = 2$) of the Weyl scalar ψ_4 at extraction radius $r_{\rm ext} = 500M$. **Right**: Trajectory of the less-massive BH.

also visually indistinguishable, as are other, higher-order ψ_4 modes.

V. CONCLUSIONS AND FUTURE WORK

`NRPyElliptic` is an extensible ID code for numerical relativity that recasts non-linear elliptic PDEs as covariant, hyperbolic PDEs. To this end it adopts the hyperbolic relaxation method of [26], in which the (pseudo)time evolution of the hyperbolic PDEs exponentially relaxes to a steady state consistent with the solution to the elliptic problem. That standard hyperbolic methods are used to solve the elliptic problem is beneficial, as consumers of numerical relativity ID generally already have expertise in solving hyperbolic PDEs. Thus the learning curve is significantly lowered for core users of `NRPyElliptic`, enabling them to build on existing expertise to modify and extend the solver.

`NRPyElliptic` leverages `NRPy+`'s reference metric infrastructure to solve the hyperbolic/elliptic PDEs in a wide variety of Cartesian-like, spherical-like, cylindrical-like, and bispherical-like coordinate systems. As its first application, in `NRPyElliptic` we solve a nonlinear elliptic PDE to set up two-puncture ID for numerical relativity. Similar to the `TwoPunctures` [20] code, `NRPyElliptic` solves the elliptic PDE for the Hamiltonian constraint in a prolate spheroidal-like coordinate system. But unlike the one-parameter coordinate system used in prolate-spheroidal coordinates or `TwoPunctures` coordinates, `NRPyElliptic` adopts `NRPy+`'s `SinhSymTP` three-parameter coordinate system, providing greater flexibility in setting up the numerical grid.

As the `SinhSymTP` numerical grid is not compactified, finite-radius boundary conditions must be applied. To address this, a new radiation boundary condition algorithm within `NRPy+` has been developed, which is based on the widely used `NewRad` radiation boundary condition driver within the `Einstein Toolkit`. While `NewRad` implements a second-order approach, `NRPyElliptic` extends to fourth and sixth finite difference orders as well. This high-order boundary condition meshes well with the high (tenth) order finite-difference representation of the elliptic operators adopted in `NRPyElliptic`.

To greatly accelerate the relaxation, we set the wavespeed of the hyperbolic PDEs to grow in proportion to the grid spacing. As the `SinhSymTP` grid spacing grows exponentially with distance from the central region of the coordinate system, so does the relaxation wavespeed. Thus this approach is many orders of magnitude faster than the traditional, constant-wavespeed choice, in fact making it fast enough to set up high-quality, full-3D BBH ID for numerical relativity.

Although `NRPyElliptic` currently requires only a tiny fraction of the total runtime of a typical NR BBH merger calculation, it is roughly 12x slower than `TwoPunctures`

when setting up the full-3D BBH ID in this work. Efforts in the immediate future will in part focus on improving this performance. To this end, a couple of ideas come to mind. First, all ID generated in this work used the trivial ($u = v = 0$) initial guess. We plan to explore whether relaxations at lower-resolutions might be used to provide a superior initial guess on finer grids, in which convergence is accelerated.

Second, due to the CFL condition, the speed of `NRPyElliptic` is proportional to the smallest grid spacing, which occurs at the foci of our `SinhSymTP` coordinate system. Typical grid spacings at the foci are $\sim 10^{-4}M$ due to extreme grid focusing there, which in turn are $\sim$1/100 those typically used in (near-equal-mass) binary puncture evolutions, indicating that a significant speed-up may be possible if superior grid structures are used.

To this end, we plan to adopt the same seven-grid bispheres grids infrastructure adopted by the `NRPy+`-based `BlackHoles@Home` [52] project. As illustrated in Fig. 9, seven-grid BiSpheres consists of seven overlapping spherical-like and Cartesian-like grids. This approach places Cartesian-like AMR grid patches over regions where the spherical-like grids would otherwise experience extreme grid focusing ($r \to 0$), constraining the smallest grid spacings in the strong-field region to $\approx M/200$. Thus with such grids, accounting for needed inter-grid interpolations, we might expect roughly a $\sim$10x increase in speed—making `NRPyElliptic` comparable in performance to `TwoPunctures` for near-equal-mass-ratio systems—all while maintaining excellent resolution in the strong-field region. Further, the Cartesian-like AMR patches on these grids are centered precisely at the locations of the compact objects, making them efficiently tunable to higher mass ratios, unlike `SinhSymTP` or other prolate spheroidal-like coordinates mentioned in this work. Extending `NRPyElliptic` to higher mass ratios in this way, as well as to other types of NR ID, will be explored in forthcoming papers.

ACKNOWLEDGMENTS

The authors would like to thank S. Brandt and R. Haas for useful discussions and suggestions during the preparation of `NRPyElliptic`. Support for this work was provided by NSF awards OAC-2004311, PHY-1806596, PHY-2110352, as well as NASA awards ISFM-80NSSC18K0538 and TCAN-80NSSC18K1488. Computational resources were in part provided by West Virginia University's Thorny Flat high-performance computing cluster, funded by NSF MRI Award 1726534 and West Virginia University. T.P.J. acknowledges support from West Virginia University through the STEM Graduate Fellowship.

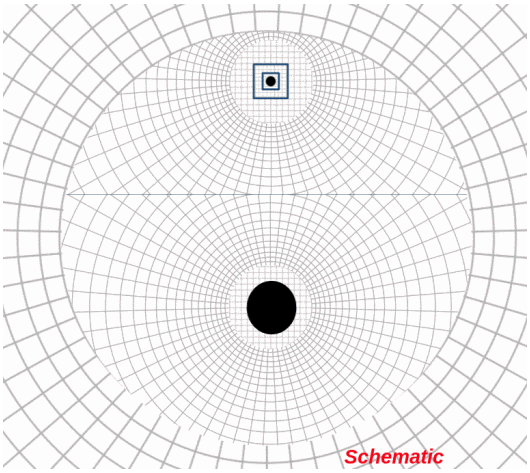

FIG. 9. Schematic of seven-grid BiSpheres numerical meshes used for `BlackHoles@Home` $\sim$6:1 mass-ratio BBH simulations during the long inspiral phase. BHs are represented as black dots.

Appendix A: Sommerfeld Boundary Conditions

Sommerfeld boundary conditions (BCs)—also referred to as radiation or transparent boundary conditions—aim to enable outgoing wave fronts to pass through the boundaries of a domain with minimal reflection. Sommerfeld BCs typically assume that for large values of r any given field $f = f(t, r)$ behaves as an *outgoing* spherical wave, with an asymptotic value f_0 as $r \to \infty$. Following `NewRad` [39], our *ansatz* for $f(t, r)$ on the boundary takes the form

$$f = f_0 + \frac{w(r - ct)}{r} + \frac{K}{r^n}\,, \tag{A1}$$

where $w(r-ct)/r$ represents an outgoing wave that solves the wave equation in spherical symmetry,[9] and K is a constant. The $1/r^n$ correction term encapsulates higher-order corrections with $n > 1$ fall-off.

We follow the hyperbolic relaxation method of [26] and `NewRad`, and apply Sommerfeld boundary conditions not to f directly, but to $\partial_t f$:

$$\partial_t f = -c\frac{w'(r - ct)}{r}\,. \tag{A2}$$

To better understand the $w'(r - ct)$ term, we compute the radial partial derivative of f as well:

$$\partial_r f = \frac{w'(r - ct)}{r} - \frac{w(r - ct)}{r^2} - n\frac{K}{r^{n+1}}\,. \tag{A3}$$

Solving Eq. (A3) for $w'(r - ct)$ and substituting into Eq. (A2) yields

$$\partial_t f = -c\left[\partial_r f + \frac{w(r - ct)}{r^2} + n\frac{K}{r^{n+1}}\right]\,. \tag{A4}$$

[9] That is, $\partial_t^2(rw) - c^2\partial_r^2(rw) = 0$.

To take care of the (as-yet) unknown $w(r-ct)/r^2$ term, notice that our *ansatz* Eq. (A1) implies

$$\frac{w(r-ct)}{r^2} = \frac{f-f_0}{r} - \frac{K}{r^{n+1}}\,, \tag{A5}$$

which when inserted into Eq. (A4) yields

$$\partial_t f = -c\left[\partial_r f + \frac{f-f_0}{r}\right] + \frac{k}{r^{n+1}}\,, \tag{A6}$$

where $k=-cK(n-1)$ is just another constant. Thus we have derived the desired boundary condition

$$\partial_t f = -\frac{c}{r}\left[r\partial_r f + (f-f_0)\right] + \frac{k}{r^{n+1}}\,. \tag{A7}$$

To generalize Eq. (A7) to arbitrary curvilinear coordinate systems $x^i_{\rm Curv}$, we make use of the chain rule

$$\frac{\partial f(x^i_{\rm Curv})}{\partial r} = \left(\frac{\partial x^i_{\rm Curv}}{\partial r}\right)\left(\frac{\partial f}{\partial x^i_{\rm Curv}}\right), \tag{A8}$$

which can be plugged into Eq. (A7) to give us Eq. (30)

$$\partial_t f = -\frac{c}{r}\left[r\frac{\partial x^i_{\rm Curv}}{\partial r}\frac{\partial f}{\partial x^i_{\rm Curv}} + (f-f_0)\right] + \frac{k}{r^{n+1}}\,. \tag{A9}$$

Returning to the original *ansatz* (Eq. A1), we would generally expect the lowest-order correction to be one order higher than the dominant, $1/r$ falloff. As the correction term in Eq. (A1) has $1/r^n$ falloff, we therefore set $n=2$ to obtain our final expression for imposing outgoing radiation boundary conditions any given field f:

$$\boxed{\partial_t f = -\frac{c}{r}\left[r\frac{\partial x^i_{\rm Curv}}{\partial r}\frac{\partial f}{\partial x^i_{\rm Curv}} + (f-f_0)\right] + \frac{k}{r^3}}\,. \tag{A10}$$

Regarding numerical implementation of this expression a couple of subtleties arise. First, note that $\partial x^i_{\rm Curv}/\partial r$ may be impossible to compute analytically, as the spherical radius r is generally easy to write in terms of the curvilinear coordinates $x^i_{\rm Curv}$, but not its inverse $x^i_{\rm Curv}(r)$.

To address this, for all coordinate systems $x^i_{\rm Curv}$ implemented in `NRPy+`, the function

$$x^i_{\rm Sph} = \Big(r(x^i_{\rm Curv}), \theta(x^i_{\rm Curv}), \phi(x^i_{\rm Curv})\Big)\,, \tag{A11}$$

is explicitly defined. If we define the Jacobian

$$J^j_{\ i} = \frac{\partial x^j_{\rm Sph}}{\partial x^i_{\rm Curv}}\,, \tag{A12}$$

and use `NRPy+` functions to invert this matrix, we obtain exact expressions for the inverse Jacobian matrix, which encodes $\partial x^i_{\rm Curv}/\partial x^j_{\rm Sph}$:

$$\left(J^{-1}\right)^i_{\ j} = \frac{\partial x^i_{\rm Curv}}{\partial x^j_{\rm Sph}}\,. \tag{A13}$$

From this, we can express $\partial x^i_{\rm Curv}/\partial r$ exactly for any curvilinear coordinate system implemented within `NRPy+`.

The second subtlety lies in formulating a way to approximate k. If the function f represented only an outgoing spherical wave, then it would exactly satisfy the advection equation

$$\left[\frac{\partial f}{\partial t}\right]_{\rm adv} \equiv -\frac{c}{r}\left[r\frac{\partial x^i_{\rm Curv}}{\partial r}\frac{\partial f}{\partial x^i_{\rm Curv}} + (f-f_0)\right], \tag{A14}$$

which is identical to Eq. (A10) but with $k=0$.

Next consider an interior point $r_{\rm int}$ directly adjacent to the outer boundary. Then, $f(r_{\rm int})$ approximately satisfies *both* the time evolution equation (e.g. Eq. 26), and the advection equation Eq. (A14). We compute $\partial_t f(r_{\rm int})$ for a given field f directly from evaluating the corresponding right-hand side of Eq. (21), and $[\partial_t f]_{\rm adv}(r_{\rm int})$ from Eq. (A14). The difference of these two equations yields the departure from the expected purely outgoing wave behavior at that point $k/r_{\rm int}^{n+1}$. From this we can immediately extract k:

$$\boxed{k = r_{\rm int}^3\left(\frac{\partial f}{\partial t} - \left[\frac{\partial f}{\partial t}\right]_{\rm adv}\right)_{\rm int}}\,, \tag{A15}$$

where again we impose $n=2$.

Our numerical implementation of Sommerfeld BCs evaluates $\partial f/\partial x^i_{\rm Curv}$ in Eq. (A14) using either centered or fully upwinded finite-difference derivatives as needed to ensure finite-difference stencils do not reach out of bounds. Unlike `NewRad`, which only implements second-order finite-difference derivatives for $\partial f/\partial x^i_{\rm Curv}$, our implementation supports second, fourth, and sixth-order finite differences.

We validated this Sommerfeld boundary condition algorithm against `NewRad` for the case of a scalar wave propagating across a 3D Cartesian grid, choosing second-order finite-difference derivatives in our algorithm. We achieved roundoff-level agreement for the wave propagating toward each of the individual faces.

Appendix B: Spatially-dependent wavespeed and relaxation-wave-crossing time

At every point in the domain we compute the smallest proper distance between neighboring points, $\Delta S_{\rm min}$, and define a local wavespeed that is proportional to the grid spacing as

$$c(\vec{x}) = \mathcal{C}_0\frac{\Delta S_{\rm min}(\vec{x})}{\Delta t}\,, \tag{B1}$$

where Δt is the time step used by the time integrator, and $\mathcal{C}_0$ is the CFL factor.

For the sake of readability, we repeat here the relationship between `SinhSymTP` coordinates and Cartesian

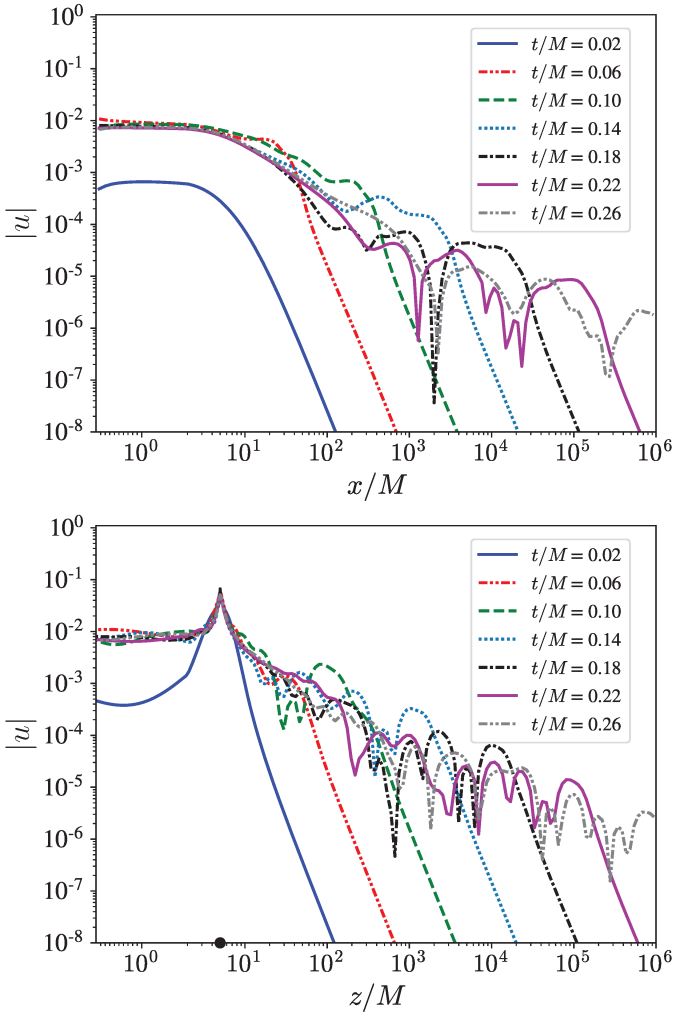

FIG. 10. **Top**: Propagation of u along gridpoints closest to the x-axis during the first relaxation-wave-crossing time for the 3D ID. **Bottom**: Same as top panel, except for gridpoints closest to the z-axis.

coordinates,

$$\begin{aligned} x &= \tilde{r}\sin(\mathtt{x}_2)\cos(\mathtt{x}_3)\,, \\ y &= \tilde{r}\sin(\mathtt{x}_2)\sin(\mathtt{x}_3)\,, \\ z &= \left(\tilde{r}^2+b^2\right)^{1/2}\cos(\mathtt{x}_2)\,. \end{aligned} \tag{34}$$

To establish the time scales associated with the propagation of relaxation waves, we calculate the relaxation-wave-crossing time along the x-axis—$(\mathtt{x}_2,\mathtt{x}_3)=(\pi/2,0)$ in Eqs. (34)—and the z-axis. `NRPyElliptic` adopts topologically Cartesian, cell-centered grids with inter-cell spacing of $\Delta\mathtt{x}_1$, $\Delta\mathtt{x}_2$, and $\Delta\mathtt{x}_3$ in the $\mathtt{x}_1$, $\mathtt{x}_2$, and $\mathtt{x}_3$-directions, respectively. Because of this, the closest points to the x-axis are obtained by setting $\mathtt{x}_2 = (\pi+\Delta\mathtt{x}_2)/2$ and $\mathtt{x}_3 = \Delta\mathtt{x}_3/2$. Likewise, the closest points to the z-axis are given by setting $\mathtt{x}_2 = \Delta\mathtt{x}_2/2$ and $\mathtt{x}_3$ to any fixed value. The relaxation-wave crossing time along the x-axis is given by

$$T^{(x)}_{\rm RC} = \int \frac{dx}{c(x)} = \int_{\mathtt{x}_1=0}^{1} \frac{f(\mathtt{x}_1)d\mathtt{x}_1}{c^{(x)}(\mathtt{x}_1)}\,, \tag{B2}$$

where

$$f(\mathtt{x}_1) = \partial_{\mathtt{x}_1} x(\mathtt{x}_1,\mathtt{x}_2,\mathtt{x}_3)\Big|_{\mathtt{x}_2=(\pi+\Delta\mathtt{x}_2)/2,\mathtt{x}_3=\Delta\mathtt{x}_3/2}\,, \tag{B3}$$

and $c^{(x)}$ is the wavespeed along the x-axis. For the choice of grid parameters described in Sec. III, the smallest grid spacing along the x-axis is given by the proper distance in the direction of the $\mathtt{x}_2$ coordinate, so that

$$c^{(x)}(\mathtt{x}_1) = \mathcal{C}_0\frac{\Delta S^{(2)}(\mathtt{x}_1)}{\Delta t} = \frac{\mathcal{C}_0\Delta\mathtt{x}_2}{\Delta t}h^{(2)}(\mathtt{x}_1)\,, \tag{B4}$$

where $h^{(2)}(\mathtt{x}_1) = \sqrt{\tilde{r}^2(\mathtt{x}_1)+b^2\cos^2(\Delta\mathtt{x}_2/2)}$ is the scale factor. Substituting Eq. (B4) in Eq. (B2) and integrating yields

$$\begin{aligned} T^{(x)}_{\rm RC} = \frac{\Delta t}{\mathcal{C}_0\Delta\mathtt{x}_2}\ &\mathrm{arctanh}\left(\frac{\mathcal{A}}{\sqrt{\mathcal{A}^2+b^2\cos^2(\Delta\mathtt{x}_2/2)}}\right) \\ &\times\sin(\Delta\mathtt{x}_2/2)\cos(\Delta\mathtt{x}_3/2)\,. \end{aligned} \tag{B5}$$

Similarly, the wavespeed along the z-axis is given by

$$c^{(z)}(\mathtt{x}_1) = \mathcal{C}_0\frac{\Delta S^{(3)}(\mathtt{x}_1)}{\Delta t} = \frac{\mathcal{C}_0\Delta\mathtt{x}_3}{\Delta t}h^{(3)}(\mathtt{x}_1)\,, \tag{B6}$$

where $h^{(3)}(\mathtt{x}_1) = \tilde{r}(\mathtt{x}_1)\sin(\Delta\mathtt{x}_2/2)$ is the scale factor in the direction of $\mathtt{x}_3$. Thus, the relaxation-wave-crossing time along the z-axis can be computed as

$$\begin{aligned} T^{(z)}_{\rm RC} &= \int \frac{dz}{c(z)} = \int_{\mathtt{x}_1=0}^{1} \frac{g(\mathtt{x}_1)d\mathtt{x}_1}{c^{(z)}(\mathtt{x}_1)} \\ &= \frac{\Delta t}{\mathcal{C}_0\Delta\mathtt{x}_3}\ \mathrm{arctanh}\left(\frac{\mathcal{A}}{\sqrt{\mathcal{A}^2+b^2}}\right)\cot(\Delta\mathtt{x}_2/2)\,, \end{aligned} \tag{B7}$$

where we used

$$g(\mathtt{x}_1) = \partial_{\mathtt{x}_1} z(\mathtt{x}_1,\mathtt{x}_2,\mathtt{x}_3)\Big|_{\mathtt{x}_2=\Delta\mathtt{x}_2/2}\,. \tag{B8}$$

Plugging in the values of the grid parameters $\mathcal{A}$, w, and step sizes, we find $T^{(x)}_{\rm RC} = 0.248M$ $(0.574M)$ and $T^{(z)}_{\rm RC} = 1.28M$ $(1.26M)$ for full-3D (axisymmetric) ID.

From Eq. (B6) we find that for fixed values of $\mathtt{x}_1$ and small angles $\mathtt{x}_2$ the wavespeed along the z-axis behaves as

$$c^{(z)} \sim \tilde{r}(\mathtt{x}_1)\,\mathtt{x}_2\,\Delta\mathtt{x}_3\,. \tag{B9}$$

Our cell-centered grid has $\mathtt{x}_{2\mathtt{i}_2} = (\mathtt{i}_2+1/2)\Delta\mathtt{x}_2$ with $\mathtt{i}_2 = 0,1,2,\ldots,\mathtt{N}_2$, and thus the wavespeed quickly increases as we move away from the axis. Such rapidly-moving signals slightly off the z-axis are not considered in our analytic estimates, yet influence the RCT so that in our full-3D simulations we observe the same value of $T_{\rm RC}\approx 0.25M\approx T^{(x)}_{\rm RC}$ along *both* axes, as shown in Fig. 10. Thus we use $T_{\rm RC} = T^{(x)}_{\rm RC}$ in all figures.

[1] R. Abbott *et al.* (LIGO Scientific, VIRGO), (2021), arXiv:2108.01045 [gr-qc].
[2] T. Nakamura, K.-I. Oohara, and Y. Kojima, Prog. Theor. Phys. Suppl. **90**, 1 (1987).
[3] M. Shibata and T. Nakamura, Phys. Rev. D **52**, 5428 (1995).
[4] T. W. Baumgarte and S. L. Shapiro, Phys. Rev. D **59**, 024007 (1999), arXiv:gr-qc/9810065.
[5] F. Pretorius, Class. Quant. Grav. **22**, 425 (2005), arXiv:gr-qc/0407110.
[6] C. Bona, T. Ledvinka, C. Palenzuela, and M. Zacek, Phys. Rev. D **67**, 104005 (2003), arXiv:gr-qc/0302083.
[7] S. Bernuzzi and D. Hilditch, Phys. Rev. D **81**, 084003 (2010), arXiv:0912.2920 [gr-qc].
[8] D. Alic, C. Bona-Casas, C. Bona, L. Rezzolla, and C. Palenzuela, Phys. Rev. D **85**, 064040 (2012), arXiv:1106.2254 [gr-qc].
[9] D. Alic, W. Kastaun, and L. Rezzolla, Phys. Rev. D **88**, 064049 (2013), arXiv:1307.7391 [gr-qc].
[10] N. Sanchis-Gual, P. J. Montero, J. A. Font, E. Müller, and T. W. Baumgarte, Phys. Rev. D **89**, 104033 (2014), arXiv:1403.3653 [gr-qc].
[11] A. M. Knapp, E. J. Walker, and T. W. Baumgarte, Phys. Rev. D **65**, 064031 (2002), arXiv:gr-qc/0201051.
[12] G. B. Cook, Living Rev. Rel. **3**, 5 (2000), arXiv:gr-qc/0007085.
[13] K. Uryū, F. Limousin, J. L. Friedman, E. Gourgoulhon, and M. Shibata, Phys. Rev. D **80**, 124004 (2009), arXiv:0908.0579 [gr-qc].
[14] K. Uryū and A. Tsokaros, Phys. Rev. D **85**, 064014 (2012), arXiv:1108.3065 [gr-qc].
[15] W. E. East, F. M. Ramazanoğlu, and F. Pretorius, Phys. Rev. D **86**, 104053 (2012), arXiv:1208.3473 [gr-qc].
[16] N. Moldenhauer, C. M. Markakis, N. K. Johnson-McDaniel, W. Tichy, and B. Brügmann, Phys. Rev. D **90**, 084043 (2014), arXiv:1408.4136 [gr-qc].
[17] E. Gourgoulhon, P. Grandclement, K. Taniguchi, J.-A. Marck, and S. Bonazzola, Phys. Rev. D **63**, 064029 (2001), arXiv:gr-qc/0007028.
[18] K. Taniguchi, E. Gourgoulhon, and S. Bonazzola, Phys. Rev. D **64**, 064012 (2001), arXiv:gr-qc/0103041.
[19] K. Taniguchi and E. Gourgoulhon, Phys. Rev. D **66**, 104019 (2002), arXiv:gr-qc/0207098.
[20] M. Ansorg, B. Bruegmann, and W. Tichy, Phys. Rev. D **70**, 064011 (2004), arXiv:gr-qc/0404056.
[21] N. Tacik, F. Foucart, H. P. Pfeiffer, C. Muhlberger, L. E. Kidder, M. A. Scheel, and B. Szilágyi, Classical and Quantum Gravity **33**, 225012 (2016), arXiv:1607.07962 [gr-qc].
[22] W. Tichy, A. Rashti, T. Dietrich, R. Dudi, and B. Brügmann, Phys. Rev. D **100**, 124046 (2019), arXiv:1910.09690 [gr-qc].
[23] A. Rashti, F. M. Fabbri, B. Brügmann, S. V. Chaurasia, T. Dietrich, M. Ujevic, and W. Tichy, (2021), arXiv:2109.14511 [gr-qc].
[24] L. J. Papenfort, S. D. Tootle, P. Grandclément, E. R. Most, and L. Rezzolla, Phys. Rev. D **104**, 024057 (2021), arXiv:2103.09911 [gr-qc].
[25] W. Barreto, P. C. M. Clemente, H. P. de Oliveira, and B. Rodriguez-Mueller, General Relativity and Gravitation **50**, 71 (2018), arXiv:1803.05833 [gr-qc].
[26] H. R. Rüter, D. Hilditch, M. Bugner, and B. Brügmann, Phys. Rev. D **98**, 084044 (2018), arXiv:1708.07358 [gr-qc].
[27] A. Meurer, C. P. Smith, M. Paprocki, O. Čertík, S. B. Kirpichev, M. Rocklin, A. Kumar, S. Ivanov, J. K. Moore, S. Singh, *et al.*, PeerJ Computer Science **3**, e103 (2017).
[28] I. Ruchlin, Z. B. Etienne, and T. W. Baumgarte, Phys. Rev. D **97**, 064036 (2018), arXiv:1712.07658 [gr-qc].
[29] "NRPy+'s webpage," `http://nrpyplus.net/`.
[30] J. W. York Jr., Phys. Rev. Lett. **82**, 1350 (1999), arXiv:gr-qc/9810051.
[31] G. Lovelace, M. Boyle, M. A. Scheel, and B. Szilagyi, Class. Quant. Grav. **29**, 045003 (2012), arXiv:1110.2229 [gr-qc].
[32] M. Alcubierre, *Introduction to 3+1 Numerical Relativity* (Oxford University Press, Oxford, U.K., 2008).
[33] F. Loffler *et al.*, Class. Quant. Grav. **29**, 115001 (2012), arXiv:1111.3344 [gr-qc].
[34] "Einstein toolkit home page," `http:einsteintoolkit.org`.
[35] "Cactus web site," `http://www.cactuscode.org`.
[36] J. M. Bowen and J. W. York Jr., Phys. Rev. D **21**, 2047 (1980).
[37] S. Brandt and B. Bruegmann, Phys. Rev. Lett. **78**, 3606 (1997), arXiv:gr-qc/9703066.
[38] E. Schnetter, Class. Quantum Grav. **27**, 167001 (2010), arXiv:1003.0859 [gr-qc].
[39] M. Alcubierre, B. Bruegmann, P. Diener, M. Koppitz, D. Pollney, E. Seidel, and R. Takahashi, Phys. Rev. D **67**, 084023 (2003), arXiv:gr-qc/0206072.
[40] P. Moon and D. E. Spencer, *Field theory handbook: including coordinate systems, differential equations and their solutions* (Springer, 2012).
[41] B. P. Abbott *et al.* (LIGO Scientific, Virgo), Phys. Rev. Lett. **116**, 061102 (2016), arXiv:1602.03837 [gr-qc].
[42] B. Wardell, I. Hinder, and E. Bentivegna, "Simulation of GW150914 binary black hole merger using the Einstein Toolkit," (2016).
[43] D. Pollney, C. Reisswig, E. Schnetter, N. Dorband, and P. Diener, Phys. Rev. D **83**, 044045 (2011), arXiv:0910.3803 [gr-qc].
[44] E. Schnetter, S. H. Hawley, and I. Hawke, Class. Quant. Grav. **21**, 1465 (2004), arXiv:gr-qc/0310042.
[45] J. Thornburg, Class. Quant. Grav. **21**, 743 (2004), arXiv:gr-qc/0306056.
[46] O. Dreyer, B. Krishnan, D. Shoemaker, and E. Schnetter, Phys. Rev. D **67**, 024018 (2003), arXiv:gr-qc/0206008.
[47] T. Goodale, G. D. Allen, G. Lanfermann, J. Massó, T. Radke, E. Seidel, and J. Shalf, in *Vector and Parallel Processing - VECPAR'2002, 5th International Conference, Lecture Notes in Computer Science* (Springer, Berlin, 2003).
[48] J. D. Brown, P. Diener, O. Sarbach, E. Schnetter, and M. Tiglio, Phys. Rev. D **79**, 044023 (2009), arXiv:0809.3533 [gr-qc].
[49] S. Husa, I. Hinder, and C. Lechner, Comput. Phys. Commun. **174**, 983 (2006), arXiv:gr-qc/0404023.
[50] M. Thomas and E. Schnetter, in *Grid Computing (GRID), 2010 11th IEEE/ACM International Confer-*

ence on (2010) arXiv:1008.4571 [cs.DC].
[51] V. Paschalidis, Z. B. Etienne, R. Gold, and S. L. Shapiro, (2013), arXiv:1304.0457 [gr-qc].
[52] "Blackholes@home home page," http://blackholesathome.net/.