The Dissertation Committee for Paul Joseph Walter

certifies that this is the approved version of the following dissertation:

Using *openGR* for Numerical Relativity

Committee:

Richard Matzner, Supervisor

Duane Dicus

Manfred Fink

Pawan Kumar

Phil Morrison

Using *openGR* for Numerical Relativity

by

Paul Joseph Walter, B.S.

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

December 2009

Acknowledgments

openGR was developed over time by a team of researchers and the author would like to acknowledge their efforts. Matt Anderson began the development of *openGR*, and was followed by the author, Jon Allen, and, in particular, Andrea Nerozzi. The author would also like to thank Uli Sperhake for providing scripts related to wave extraction. During this time, the author's advisor Richard Matzner has overseen our collective work, providing the author with insights and direction.

openGR is a *SAMRAI*-based (Structured Adaptive Mesh Refinement Application Infrastructure) code, and thus its development would not be possible without the work done by the *SAMRAI* team at the Lawrence Livermore National Laboratory (LLNL). All visualization of *openGR* output was created using *VisIt*.

The author acknowledges the Texas Advanced Computing Center (TACC) at The University of Texas at Austin for providing HPC and visualization resources that have contributed to the research results reported within this work. URL: http://www.tacc.utexas.edu. All simulations reported here were carried out on Ranger (Spur for visualization) at TACC using Teragrid account TG-PHY090030.

Paul Joseph Walter

The University of Texas at Austin

December 2009

Using *openGR* for Numerical Relativity

Publication No. _____

Paul Joseph Walter, Ph.D.
The University of Texas at Austin, 2009

Supervisor: Richard Matzner

Binary black hole mergers are the strongest expected producers of gravitational radiation in the universe. Ground-based and proposed space-based gravitational wave detectors will benefit from simulations modeling the mergers and extracting the resulting gravitational waveforms. Producing templates of waveforms will both aid the likelihood of detection and the estimation of parameters (mass ratio, spin, etc.). *openGR* is a modular, open framework developed to carry out simulations of binary black hole mergers. While designed with the two-body problem in mind, *openGR* will evolve most general spacetimes.

This work overviews the capabilities of *openGR* and the correspond-

ing physics involved. *openGR* supports both excision and puncture methods. When excising the black hole, to date we have used only the weakly hyperbolic ADM formulation of the Einstein's equations. As expected from a weakly hyperbolic system, instabilites arise and crash the code when simulating even just a single boosted black hole in Kerr-Schild coordinates. In contrast, successful mergers of two black holes have been achieved using the puncture method. We demonstrate such a simulation in Ch 8. In this case, we make use of a BSSN formulation of Einstein's equations (a strongly hyperbolic system).

Contents

List of Tables

List of Figures

Chapter 1

Introduction

openGR is a framework to support large numerical simulations in general relativity. It has been developed to solve the particular problem of simulating binary black hole mergers and extracting the resulting gravitational radiation. Nonetheless, *openGR* is capable of evolving any general relativistic spacetime. As its name suggests, *openGR* is an open source code available for download at http://wwwrel.ph.utexas.edu/openGR. Binary black hole mergers are the largest expected emitters of gravitational waves in the universe. Two compact objects in circular orbit produce gravitational waves with wave strain h of

$$h = \frac{16\mu v^2}{r}, \tag{1.1}$$

where μ is the reduced mass of the system, v is the average tangential velocity, and r is the distance the detector is away from the sources (using geometrics units where $G = c = 1$). This estimate (Eq. 1.1) is based on the quadrupole

approximation to gravitational radiation. Having smaller radii than other objects of the same mass, black holes are allowed to get closer to each other prior to merger and thus have higher tangential speeds. They are thus the potentially strongest possible source of gravitational waves.

Gravitational waves are being searched for using ground based interferometers, such as LIGO (Laser Interferometer Gravitational Wave Observatory), Virgo, GEO600, and TAMA300 [10, 11, 12, 13]. Given the distance from the sources, the wave strain of the gravitional wave will be very weak when reaching the detectors. Numerical Relativity plays a role in aiding detection by providing templates of the expected gravitational wave signals. Match filtering between the detector data and the numerical waveform templates allows for a significant increase in the signal-to-noise ratio, increasing the probability of detection. The numerical templates are also important for parameter estimation (such as the mass-ratio and spin) of sources.

In 2005, Frans Pretorius was the first to successfully simulate a full orbit of black holes before merger and extracting the gravitational radiation [14]. His code made use of Generalized Harmonic Coordinates. Shortly after, two groups (University of Texas at Brownsville and NASA Goddard) both achieved simular success using an approach based on the BSSN formulation with moving punctures (a method *openGR* now employs) [15, 16]. Other groups have since followed suit.

An initial application of *openGR* will be to provide another code in the Numerical Relativity community for code verification purposes (checking

that we get the same results), and has the hope of being a more efficient system. One difference between *openGR* and similar finite-difference codes being employed in Numerical Relativity is that *openGR* is based on SAMRAI (Structure Adaptive Mesh Refinement Application Infrastructure) while most other codes are based on *Cactus*. This difference in infrastructure will possibly lead to performance differences as the community moves to larger simulations. The finite-differencing scheme is fourth order accurate (second order at the boundaries). *openGR* has been developed during the past decade at the University of Texas at Austin [1].

1.1 Two Tracks

Throughout this work, two different track are discussed based on the history and capabilities of *openGR*. Initially, *openGR* excised the black holes from the computational domain (Ch. 4). Development of certain capabilities were based on this approach. Due to our inability to overcome instabilities when using excision, we have since shifted to using the puncture method as our approach (Ch. 4). The puncture method has proven to be sucessful for other Numerical Relativity groups and is computationally simpler in that it does not require excising the black holes from the spacetime.

Each method (excision and puncture) has its own initial data, formulations of the Einstein equations, and gauge conditions. Sections of this work that deal only with excision are marked by "*Excision:*"), such as inner boundary conditions (Sec. 6.2) as well as spheroidal overlapping coordinate patches

(Sec. 2.2.2). In a similar fashion, sections involving only the puncture method are marked by "*Puncture*".

Common to both approaches are the overview of *openGR*'s capabilities (Ch. 2), outer boundary conditions (Sec. 6.1), and the extraction of gravitational waves (Ch. 7). Results from the scaling and performance of *openGR* simulations (Ch. 8) come strictly from using the puncture approach.

1.2 Conventions

- Metric signature: $(-,+,+,+)$
- Greek indices $(\mu,\nu,...)$ span $0,1,2,3$.
- Latin indices (i,j,k) span $1,2,3$.
- Geometric units: $G=1$, $c=1$
  - Distances and times are given in terms of mass M, e.g.:
    - Distance: $d=M \Rightarrow d=\frac{GM}{c^2}$
    - Time: $t=M \Rightarrow t=\frac{GM}{c^3}$
- Covariant derivative operator: ∇
- Determinant of the 3-metric g_{ij}: g
- Determinant of the 4-metric $g_{\mu\nu}$: $\mathbf{g}$

Chapter 2

Overview of Capabilites

2.1 The Computational Framework

In this section we discuss the computational framework upon which *openGR* is built, which are the *SAMRAI* library and the solver packages *PVODE* and *KINSOL* used for solving the hyperbolic (time evolution) and elliptic (constraint) equations, respectively. The two packages *PVODE* and *KINSOL* are independent from *SAMRAI* and written in the programming language C. Fortunately, *SAMRAI* comes with C++ wrappers for both packages to embed them into the *SAMRAI* framework.

2.1.1 The SAMRAI Library

The *SAMRAI* library provides an adaptive mesh refinement framework for numerical simulations [18]. The computational grid is implemented as a col-

lection of structured grid components. The adaptive mesh refinement structure is a hierarchy of levels with different resolutions nested within the *SAMRAI* framework. Each level is divided in a series of rectangular patches that are assigned to the different processors used in the numerical simulation.

The intrinsic C++ design of the *SAMRAI* library provides a modular design with flexibility in terms of whether to use features as a "black box" or make modifications to a specific application. All the fundamental features of *SAMRAI* are defined as object oriented classes whose key functions are declared as virtual and can be inherited and overwritten by the user.

A specific example in *SAMRAI* is the refinement operation between different levels to fill the ghost zones of a finer level using interpolation from the coarser one. *SAMRAI* comes with a simple linear interpolation algorithm as default. *openGR* requires higher order interpolation since finite differences are calculated at fourth order in space. The flexibility of *SAMRAI* allows the user to inherit the class RefineOperator and overwrite the function *refine* to introduce higher order stencils for interpolation than are inherent with *SAMRAI*. Since *openGR* is fourth order accurate, it is necessary for the interpolation to be of sixth order. This way, the interpolation will not be the main source of error, but instead the accuracy of *openGR* will depend upon the (fourth) order of the finite difference stencils.

Just like the RefineOperator class, all the key ingredients of the AMR framework can be inherited and modified. For example, *SAMRAI* and *openGR* use uniform load balancing to share the workload among processors, but we

are planning to modify this feature to introduce a more adapted and optimized load balancing scheme that should improve the code efficiency. Making code improvements with *SAMRAI* is relatively straightforward.

2.1.2 *PVODE*

We use the mixed Adams-Bashforth Adams-Moulton method to solve the hyperbolic evolution equations of the Einstein equations. The method is implemented in the PVODE package of the SUNDIALS suite. This method has a variable step size and solves the initial value problem

$$\dot{y} = f(t, y), \tag{2.1}$$

$$y(t_0) = y_0. \tag{2.2}$$

The Adams-Bashforth Adams-Moulton predictor-corrector method originates from the approach where Eq. (2.1) is integrated on both sides and the integral replaced with a quadrature formula

$$y(t) - y(t_0) = \int_{t_0}^{t} f[\tau, y(\tau)]\, d\tau, \tag{2.3}$$

$$y(t) \approx y(t_i) + \sum_{j=0}^{k} A_j f[t_j, y(t_j)], \tag{2.4}$$

where $t_i \leq t_0 \leq t_1 \leq ... \leq t_k \leq t$; A_j are the appropriate quadrature coefficients.

In general a linear multistep method for the solution of Eq. (2.1) is of

the form

$$\sum_{i=0}^{q} \alpha_i y_{n+i} + \Delta t \sum_{i=0}^{q} \beta_i f\left(t_{n+i}, y_{n+i}\right) = 0, \tag{2.5}$$

where Δt is the step size. The coefficients α_i and β_i determine the method.

The Adams-Bashforth method is an explicit multistep method ($\beta_q = 0$). The other assumptions are that the only nonvanishing α coefficients are $\alpha_q = 1$ and $\alpha_{q-1} = -1$, so that Eq. (2.5) becomes

$$y_n = y_{n-1} + \Delta t \sum_{i=0}^{q-1} \beta_i f\left(t_{n+i}, y_{n+i}\right). \tag{2.6}$$

Once the order of the method q is fixed, the coefficients β_i can be determined using the Lagrange formula for polynomial interpolation, yielding

$$\beta_{q-i-1} = \frac{(-1)^i}{i!\,(q-i-1)!} \int_0^1 \prod_{i=0}^{q-1} (u+i)\, du, \tag{2.7}$$

where $i = 0, .., q-1$. For example a fourth order Adams-Bashforth method has coefficients $\beta = \left[-\frac{3}{8}, \frac{37}{24}, -\frac{59}{24}, \frac{55}{24}\right]$.

The Adams-Moulton method is similar to Adams-Bashforth, with the single difference that the assumption $\beta_q = 0$ is dropped, making the method implicit. This implies that a q-step Adams-Moulton scheme is of order $q+1$ while a q-step Adams-Bashforth scheme is of order q. The procedure for calculating the coefficients is similar and uses again the Lagrange formula for polynomial interpolation.

Being implicit, the Adams-Moulton is method more accurate than Adams-

Bashforth, but it is computationally more expensive since solving an implicit problem requires the use of more advanced numerical techniques. *PVODE* offers a mixed version of Adams-Bashforth and Adams-Moulton. This methodology is often referred to as "predictor-corrector": it uses first the Adams-Bashforth method to calculate the value of the variable y at the timestep n (predictor phase) and then uses Adams-Moulton to improve the value (corrector phase); given that the Adams-Bashforth part calculates the solution at the timestep n, the following Adams-Moulton part is no longer implicit and can be solved using functional iteration until convergence is met.

PVODE offers the possibility to use the Adams methods up to twelfth order in time. For the Adams-Bashforth method of order q to work, a number of q timesteps are required. When the simulation starts, however, only one timestep is given; this lack of information is solved by varying the order of method and size of timesteps; the simulation starts using first order Adams-Bashforth and very small timesteps and slowly increases order and timestep to reach the desired integration order.

When compared to the standard Runge-Kutta scheme used for time integration in the majority of codes used in numerical relativity, the Adams methodology, given the implicit nature of the Adams-Moulton part, proves to be more accurate and stable. Nevertheless it is more demanding in terms of memory allocation and efficiency; in fact the need to keep the information of twelve previous timesteps increases the memory storage of the program. Moreover, the variability of timesteps does now allow the use of AMR technologies

like Berger-Oliger, which would increase the efficiency of the code if it could be implemented. In our case, instead, all the AMR levels must evolve using the same timestep, which results in a less efficient than optimal code.

2.1.3 *KINSOL*

The ADM decomposition of Einstein's equations introduces a set of elliptic equations, the constraints, defined on each spatial hypersurface. Solving these constraints is necessary to have well defined initial data that are solutions of Einstein's equations. These constraints could be used to correct the evolved variables periodically, which is known as constrained evolution.

The governing equations result in a set of non-linear algebraic equations, and Newton's methods provide the simplest way to solve the system. The problem can be stated as follows: given n unknown real functions,

$$\mathbf{F}(\mathbf{x}) = \begin{bmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ . \\ . \\ . \\ f_n(\mathbf{x}) \end{bmatrix}, \tag{2.8}$$

where $\mathbf{x} = (x_1, x_2, ..., x_n)$, a vector $\mathbf{s} = (s_1, s_2, ..., s_n)$ is sought so that $\mathbf{F}(\mathbf{s}) = 0$. Newton's method for solving this equation involves iterating an initial guess, $\mathbf{x}_0$:

$$\mathbf{x}_{n+1} = \mathbf{x}_n - J\left(\mathbf{x}_n\right)^{-1} \mathbf{F}\left(\mathbf{x}_n\right), \tag{2.9}$$

where $J\left(\mathbf{x}_n\right)$ is the Jacobian of $\mathbf{F}$ at $\mathbf{x}_n$.

Newton's method for several variables, Eq. (2.9), now only requires the solution of a system of linear equations at each iteration, until the solution converges to a specified error tolerance.

The particular elliptic solver used by *KINSOL* is based on the Krylov procedure for systems of non-linear equations. The method starts with an initial guess $\mathbf{x}_0$ for the system, and after k iterations produces an approximate solution $\mathbf{x}_k$ from a Krylov space generated by a vector c,

$$\mathcal{K}\left(A, c\right) = \text{span}\left\{c, Ac, ..., A^{k-1}c\right\}. \tag{2.10}$$

KINSOL uses the Generalized Miniminum RESsidual method (GMRES), which is one of the options of the Krylov methods.

For large simulations, we have found that *KINSOL* converges too slowly ($>$ 24 hours) to be of use when solving the initial data. As a result, we are considering other options for initial data solvers, such as implementing a multigrid preconditioner for *KINSOL*. Another option is to use an initial data solver based on the spectral method developed by Marcus Ansorg ([19]).

2.2 Multiple Coordinate Patches

When excising a black hole singularity from a Cartesian computational domain, errors crop up resulting from the lego-shapped inner boundary (see Figure 2.2). To get around this, spheroidal patches have been implemented to reduce excision error by mimicking the shape of a boosted black hole. As it is often convenient for outer boundary conditions and gravitational wave extraction to have a computational domain spherical in shape, overlapping spherical patches have also been implemented. Interpolations between all patches is sixth order. The equations for the spherical and spheroidal coordinates which can be found in Ref. [1] are reproduced here for completeness.

2.2.1 Spherical Overlapping Patches

When extracting gravitational waves from a spacetime, it is convenient to do so as far as possible from the source. Since they are ripples of spacetime itself, gravitational waves need to be extracted far from the source, in the linear (weak gravity) regime where it is possible to differentiate the waves from the nearly static spacetime. Meanwhile, the simulation must have enough resolution in the radial direction (the direction of propagation of the gravitational waves) to capture the finer features of the waveforms. One way to achieve the necessary resolution at large distances is to use spherical grids, where, at a fixed resolution, the number of grid points (n) is proportional to the distance from the source (r). For a cartesian grid, $r \sim n^3$. Since spherical grids will have coordinate singularities at the poles, to cover the entire sphere we use

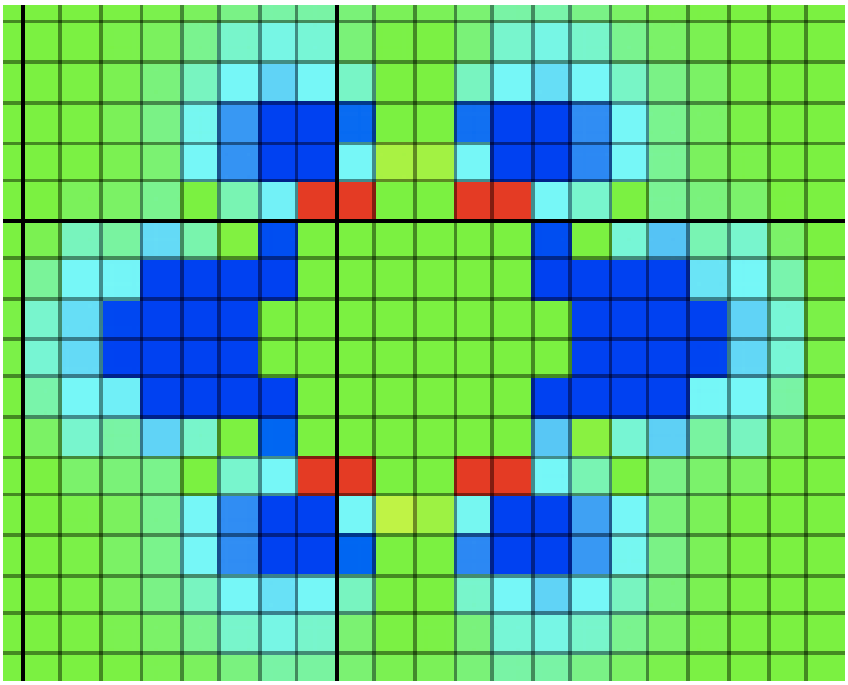


Figure 2.1: Radially excising a black hole from a cartesian domain results in a lego-shaped pattern at the inner boundary. This image is the error (difference in the numerical and analytic values) in the metric component g_{xx}. Image courtesy of Matt Anderson.

overlapping spherical grids, each with the poles cut out and one grid rotated by 90° with respect to the other. For the first spherical grid, relating the spherical coordinates (r,θ,ϕ) to cartestian coordinates (x,y,z) gives

$$\begin{aligned} x &= r\sin\theta\cos\phi \\ y &= r\sin\theta\sin\phi \\ z &= r\cos\theta \end{aligned} \tag{2.11}$$

over the domain $r \in [r_{min}, r_{max}], \theta \in [\theta_{min}, \pi - \theta_{min}], \phi \in [0, 2\pi)$. The other spherical patch is rotated by 90°. For the second spherical grid, the relation between spherical and Cartesian coordinates is

$$\begin{aligned} x &= r\cos\theta \\ y &= r\sin\theta\sin\phi \\ z &= r\sin\theta\cos\phi \end{aligned} \tag{2.12}$$

over the domain $r \in [r_{min}, r_{max}], \theta \in [\theta_{min}, \pi - \theta_{min}], \phi \in [0, 2\pi)$. The intended use of the overlapping spherical grids is to have them start (specified by r_{min}) at near the outer edges of an inner Cartesian grid (where the dynamics take place).

2.2.2 *Excision:* Spheroidal Overlapping Patches

Spheroidal overlapping coordinate patches are similar to the spherical overlapping coordinate patches, except that their shape can be distorted to mimic the shape of a boosted black hole. Thus, we use spheroidal overlapping coordinate patches which surround each black hole. The coordinate transformation from Cartesian coordinates is given by

$$\begin{aligned} x &= a \cosh r \sin\theta \cos\phi \\ y &= a \cosh r \sin\theta \sin\phi \\ z &= a \sinh r \cos\theta \end{aligned} \tag{2.13}$$

over the domain $r \in [r_{min}, r_{max}], \theta \in [\theta_{min}, \pi - \theta_{min}], \phi \in [0, 2\pi)$. The other spheroidal patch is rotated by $90\,^\circ$. In terms Cartesian coordinates this gives

$$\begin{aligned} x &= a \cosh r \cos\theta \\ y &= a \cosh r \sin\theta \sin\phi \\ z &= a \cosh r \sin\theta \cos\phi \end{aligned} \tag{2.14}$$

over the domain $r \in [r_{min}, r_{max}], \theta \in [\theta_{min}, \pi - \theta_{min}], \phi \in [0, 2\pi)$. Here, r_{min} is given by $r_{min} = \tanh^{-1} \frac{1}{\gamma}$, where $\gamma = \frac{1}{\sqrt{1-v^2}}$ and v is the speed of the black hole. a is a scale factor given by $a = \frac{r_{mask}}{\cosh r_{min}}$ where r_{mask} can be found from $x^2 + y^2 + \gamma^2 z^2 = (r_{mask})^2$. r_{mask} is the excision radius for the spacetime.

The above treatment presumed a boost in the z directon. For a more

general case, it would be necessary to have an apparent horizon finder and a scheme that fits the patches in shape (and more general equations would be necessary to fit that shape) of the apparent horizon of the black holes for cases including arbitrarily spinning, boosted, black holes.

2.2.3 Patch Layout

Putting all the patches together (overlapping spheriodal coordinate patches for each black hole, a larger Cartesian coordinate patch, and overlapping spherical patches to treat the outer boundaries) gives us a picture such as Figure 2.2.3.

As treated by *SAMRAI*, all the coordinate patches are logically rectangular. In *openGR* we have placed all the different coordinate patches on a single *SAMRAI* level (see Figure 2.2.3). We constructed a mask function that keeps track of where the different coordinate patches on the *SAMRAI* level. The spaces between coordinate patches are to allow for the outer boundary points. The initial intention of placing the different coordinate patches on a single level was to be able to alter the size of each coordinate patch as specified by the user.

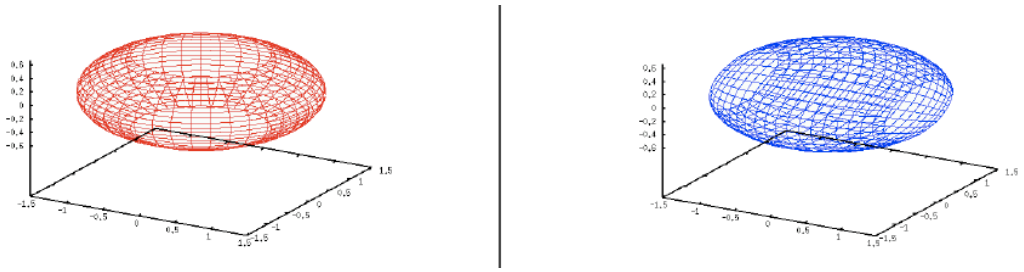


Figure 2.2: To reduce error at the inner boundary, overlapping spheroidal patches or grids are introduced to conform to the shape of black holes. Due to coordinate singularies at the poles, two overlapping grids are necessary, where each grid has its poles cut out. One grid is then rotated 90° with respect to the other, allowing for the entire spheroid to be covered. There is sixth order interpolation between the two patches. Image courtesy of Matt Anderson.

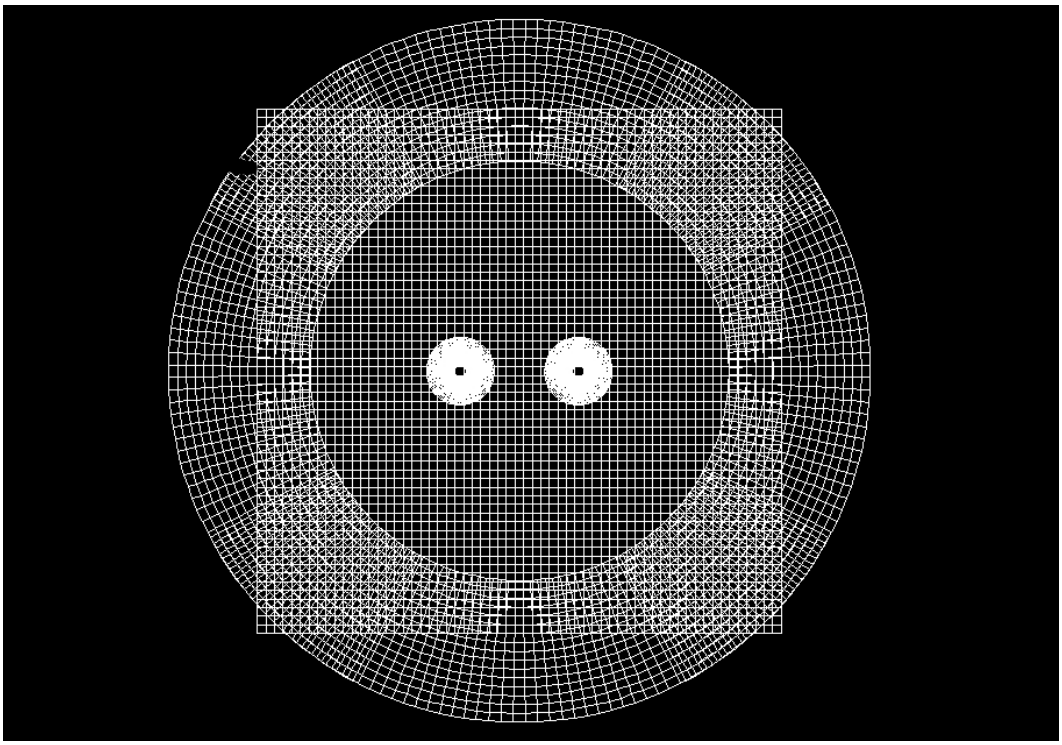


Figure 2.3: An overall view of the possibilities using multiple patches. Each black hole is excised and surrounded by two overlapping spheroidal patches. The spheroidal patches comove with the black holes inside a larger cartesian domain. Further out, overlapping spherical patches are used to treat the outer boundary and can simplify gravitational wave extraction in the wave zone. There is sixth order interpolation between all the patches. Image courtesy of Jon Allen.

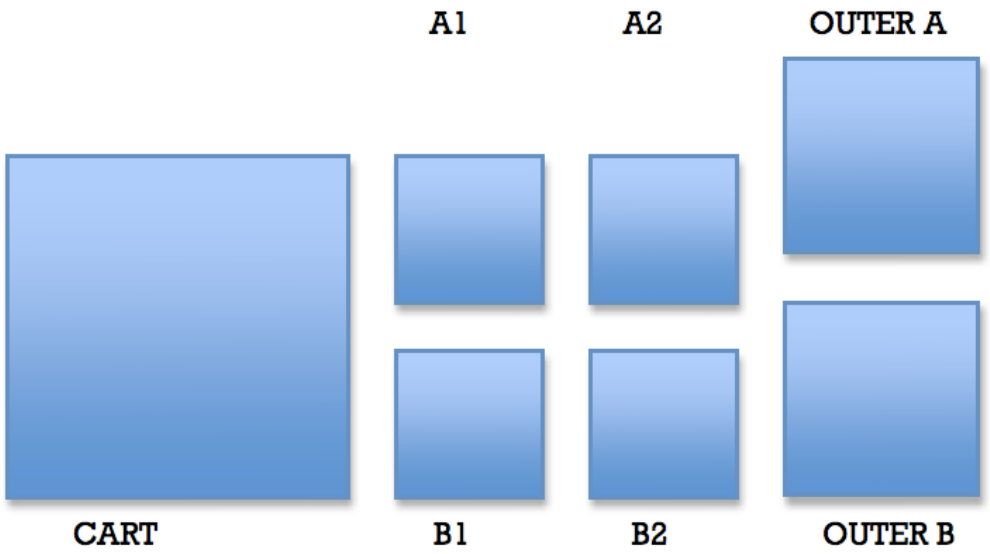


Figure 2.4: Instead of having a different patch on each level, a layout of all the patches on a single level is used. A mask function is used to keep track of the points for the various patches. Between patches, spacing is left for the outer boundary points of each patch (usually five points or cells per patch).

Chapter 3

3+1

The Einstein field equations are given by

$$G_{\mu\nu} = 8\pi\, T_{\mu\nu}, \tag{3.1}$$

where $G_{\mu\nu}$ are the components of the Einstein tensor and $T_{\mu\nu}$ are the components of the stress energy tensor. In the absense of matter,

$$G_{\mu\nu} = 0, \tag{3.2}$$

where the Einstein tensor is

$$G_{\mu\nu} = R_{\mu\nu} - \frac{1}{2}\, g_{\mu\nu}\, R. \tag{3.3}$$

Of interest is considering

$$G_{0\mu} = 0, \tag{3.4}$$

which are constraint equations. Evolution equations come from

$$G_{ij} = 0. \tag{3.5}$$

When trying to get a sense of the dynamics of the gravitational field as a function of time, the spacetime can be decomposed with a $3 + 1$ split. The ADM formulation makes clear a split between space and time.

3.1 *Excision:* ADM Formulation

The ADM formulation consists of breaking spacetime up into a foliation of spacelike hypersurfaces (Σ_t, Σ_{t+dt}, etc). The geometry from one hypersurface to the next can be constructed from the 3-metric g_{ij} residing on a hypersurface, the lapse α, and shift vector β^i. The lapse relates the proper time between hypersurfaces $d\tau$ as measured by Eulerian observers (observers moving normal to the hypersurface) to the coordinate time t by $d\tau = \alpha dt$. The shift function measures the relative velocity between lines of constant spatial coordinates and Eulerian observers

$$x^i_{t+dt} = x^i_t - \beta^i dt.$$

The lapse α and shift β^i are gauge choices determined by the choice of coordinates. The line element expression for the metric is given by

$$ds^2 = -\alpha^2 dt^2 + \gamma_{ij} \left(dx^i + \beta^i dt\right) \left(dx^j + \beta^j dt\right). \tag{3.6}$$

This gives

$$g_{\mu\nu} = \begin{bmatrix} -\alpha^2 + \beta^i\beta_i & \beta_j \\ \beta_i & \gamma_{ij} \end{bmatrix} \iff g^{\mu\nu} = \frac{1}{\alpha^2}\begin{bmatrix} -1 & \beta^j \\ \beta^i & \gamma^{ij}\alpha^2 - \beta^i\beta^j \end{bmatrix}. \quad (3.7)$$

The unit normal vector n^μ to the hypersurfaces is given by

$$n^\mu = \left(\frac{1}{\alpha}, -\frac{\beta^i}{\alpha}\right), \quad (3.8)$$

$$n_\mu = (-\alpha, 0), \quad (3.9)$$

which is the 4-velocity of Eulerian observers. Expressed in terms of the lapse and coordinate time t,

$$n^\mu = -\alpha\nabla^\mu t. \quad (3.10)$$

From the foliation of the spacetime, there is a relation for the coordinate time

$$t^\mu = \alpha n^\mu + \beta^\mu. \quad (3.11)$$

Another useful quantity is the extrinsic curvature K_{ij} which is a measure of the change in the unit normal vector n^μ as it is parallel transported along the hypersurface. It is given by

$$K_{\mu\nu} = -(\nabla_\mu n_\nu + n_\mu n^\alpha \nabla_\alpha n_\nu). \quad (3.12)$$

The components $K^{00} = K^{0i} = 0$, so from now on we will focus on K_{ij}.

3.2 Evolution Equations

Evolution equations result from the condition that $G_{ij} = 0$, providing six equations of 2nd order in the metric. This can be written as 12 first order equations as follows:

$$\begin{aligned} \partial_t g_{ij} &= -2\alpha K_{ij} + \nabla_i \beta_j + \nabla_j \beta_i, & (3.13) \\ \partial_t K_{ij} &= \alpha \left[{}^{(3)}R_{ij} + K K_{ij} - 2K_{ij} K^k{}_j \right] & (3.14) \\ &- \nabla_i \nabla_j \alpha + \beta_k \nabla^k K_{ij} + K_{ik} \nabla^k \beta_j + K_{kj} \nabla^k \beta_i. \end{aligned}$$

3.3 Constraint Equations

The constraint equations result from the condition that $G_{0\mu} = 0$ (in vacuum), which can be divided into the Hamiltonian constraint ($G_{00} = 0$) and the momentum constraints ($G_{0i} = 0$). These can be written in terms of the metric and extrinsic curvature as follows:

$$\text{Hamiltonian constraint:} \quad {}^{(3)}R + K^2 - K_{ij} K^{ij} = 0, \qquad (3.15)$$

$$\text{Momentum constraint:} \quad \nabla_j K^{ij} - \nabla_i K = 0. \qquad (3.16)$$

3.3.1 The Conformal Transverse-Traceless Decomposition

To get the constraints into a more managable form to solve, we follow the conformal transverse-traceless decomposition of York and Piran [24]. This approach converts the constraint equations into elliptic equations for a conformal factor ϕ and a vector potential w^i which can be solved for using *KINSOL*. Since the constraints are solved via an iterative process, an initial guess for the metric g_{ij} and the extrinsic curvature K_{ij} is needed. In what follows, a tilde denotes a trial field as opposed to the solved field (no tilde). For this decomposition, $K = \tilde{K}$, and so ϕ and w^i will only affect the trace-free part of the extrinsic curvature

$$A_{ij} = K_{ij} - \frac{1}{3} g_{ij} K. \tag{3.17}$$

The relationship between the solved fields, trial fields, and ϕ and w^i is given by

$$g_{ij} = \phi^4 \tilde{g}_{ij}, \tag{3.18}$$

$$A^{ij} = \phi^{-10} [\tilde{A}^{ij} + (\tilde{lw})^{ij}], \tag{3.19}$$

where

$$(\tilde{lw})^{ij} \equiv \tilde{\nabla}^i w^j + \tilde{\nabla}^j w^i - \frac{2}{3} \tilde{g}^{ij} \tilde{\nabla}_k w^k. \tag{3.20}$$

The elliptic equations to be solved for ϕ and w^i are given by

$$\tilde{\nabla}^2\phi = \frac{1}{8}\left[\tilde{R}\phi + \frac{2}{3}\tilde{K}^2\phi^5 - \phi^{-7}\left(\tilde{A}^{ij} + (\tilde{lw}^{ij}\right)\left(\tilde{A}_{ij} + (\tilde{lw})_{ij}\right)\right] \quad (3.21)$$

$$\tilde{\nabla}_j(\tilde{lw})^{ij} = \frac{2}{3}\tilde{g}^{ij}\phi^6\tilde{\nabla}_j\tilde{K} - \tilde{\nabla}_j\tilde{A}^i j. \quad (3.22)$$

Chapter 4

Excision

One way of treating singularies is to cut them out of the computational domain completely, a method known as excision. If the region excised is inside the apparent horizon of the black hole, no information *should* be able to propagate to the region outside the black hole. Due to numerical error, this is not actually the case, but spheroidal overlapping coordinate patches do improve stability. *openGR* does not yet evolve excised ADM evolutions stably. However, excision has proven to be a sucessful method for simulating the merger of binary black holes using other formaulations of General Relativity[14].

4.1 *Excision:* Kerr-Schild Coordinates

The metric for a single black hole in Kerr-Schild coordinates [25] consists of a function added to a flat background given by

$$ds^2 = \eta_{\mu\nu} dx^\mu dx^\nu + 2H l_\mu l_\nu dx^\mu dx^\nu. \tag{4.1}$$

H is a scalar function given by

$$H = \frac{Mr}{r^2 + a^2 \cos^2\theta}, \tag{4.2}$$

where a is the dimensionless spin parameter. l_μ is a null vector given by

$$l_\mu = (1, \frac{rx + ay}{r^2 + a^2}, \frac{ry - ax}{r^2 + a^2}, \frac{z}{r}), \tag{4.3}$$

where

$$\begin{aligned} z &= r\cos\theta, \\ r^2 &= \frac{1}{2}(\rho^2 - a^2) + \sqrt{\frac{1}{4}(\rho^2 - a^2)^2 + a^2 z^2}, \\ \rho &= \sqrt{x^2 + y^2 + z^2}. \end{aligned} \tag{4.4}$$

The 3-metric in Kerr-Schild coordinates is similarly given by

$$g_{ij} = \delta_{ij} + 2H l_i l_j, \tag{4.5}$$

where l_i is the spacial part of l_μ.

4.1.1 Non-spinning Black Hole

In its current state, *openGR* only allows for a single non-spinning Kerr-Schild black hole when using excision [1]. For a non-spinning Kerr-Schild black hole, the singularity resides at $r = 0$. Extension to two excised black holes (and/or spinning holes) is straight forward, but has not yet been implemented. This property is useful for excision, where the computational excision radius is placed inside the apparent horizon, but outside the singularity. Thus the singularity is excised from the computational domain. Non-spinning Kerr-Schild initial data in spherical coordinates is giving by

$$g_{ij} = \begin{bmatrix} 1 + \frac{2M}{r} & 0 & 0 \\ 0 & r^2 & 0 \\ 0 & 0 & r^2 \sin^2\theta \end{bmatrix} \tag{4.6}$$

$$K_{ij} = \begin{bmatrix} -\frac{2M(r+M)}{r^3}\sqrt{\frac{r}{r+2M}} & 0 & 0 \\ 0 & 2M\sqrt{\frac{r}{r+2M}} & 0 \\ 0 & 0 & 2M\sqrt{\frac{r}{r+2M}}\sin^2\theta \end{bmatrix} \tag{4.7}$$

The gauge conditions for the lapse and shift are found analytically from matching the Kerr-Schild metric to the general ADM metric, giving

$$\alpha = \sqrt{\frac{r}{r+2M}}, \tag{4.8}$$

$$\beta^i = \begin{bmatrix} \frac{2M}{r+2M} & 0 & 0 \end{bmatrix}. \tag{4.9}$$

In Cartesian coordinates these become

$$g_{ij} = \begin{bmatrix} 1+\frac{2Mx^2}{r^3} & \frac{2Mxy}{r^3} & \frac{2Mxz}{r^3} \\ \frac{2Mxy}{r^3} & 1+\frac{2My^2}{r^3} & \frac{2Myz}{r^3} \\ \frac{2Mxz}{r^3} & \frac{2Myz}{r^3} & 1+\frac{2Mz^2}{r^3} \end{bmatrix}, \tag{4.10}$$

$$K_{ij} = \begin{bmatrix} [Px^2-r^2]Q & PxyQ & PxzQ \\ PxyQ & [Py^2-r^2]Q & (PyzQ \\ (PxzQ & (PyzQ & [Pz^2-r^2]Q \end{bmatrix}, \tag{4.11}$$

$$\tag{4.12}$$

where

$$Q = -\frac{2M}{r^4\sqrt{1+\frac{2M}{r}}}$$

and

$$P = \left(\frac{M}{r}+2\right),$$

with the lapse and shift given by

$$\alpha = \sqrt{\frac{r}{r+2M}}, \tag{4.13}$$

$$\beta^i = \left[\frac{2Mx}{r^2} \quad \frac{2My}{r^2} \quad \frac{2Mz}{r^2}\right]. \tag{4.14}$$

4.2 Stability

Almost certainly because the ADM formulation of Einstein's equations are only weakly hyperbolic, we have not managed to have stable evolutions of a single black hole when using excision. Boosted black hole simulations crash in a time on the order of $t \sim 10M$, where M is the mass of the black hole. The higher the boost (or the greater the velocity) of the black hole, the quicker the simulation will crash due to instabilities arising on the inner boundaries of the spheroidal patches. Simulations of a single black hole will also eventually crash [1]. These instabilities moved us to the puncture method (Ch. 5). However, it should be noted that the unstable simulations were carried out on relatively small domains ($[-10M, 10M]$ in each spatial direction) with low resolutions ($\sim \frac{10 \text{ points}}{M}$). We also found some adjustments to the code that extended the lifetime of the simulations, as we discuss below.

4.2.1 Densitized Lapse

Densitizing the lapse consists of multiplying the analytically known value for the lapse (since single black hole spacetimes have exact solutions) by some

power of the determinant of the metric.

$$\alpha = \left(\frac{g}{g_{KS}}\right)^{\left(\frac{1}{n}\right)} \alpha_{KS} \tag{4.15}$$

A typical value for the power is $n = 3$. Also, g is the determinant of the numerical metric, while g_{KS} and α_{KS} are the analytic Kerr-Schild metric determinant and lapse, respectively. Densitizing the lapse has been shown to improve the hyperbolicity of the ADM formulation, but it still remains weakly hyperbolic [26].

4.2.2 Constrained Evolution

Constrained evolution consists of periodically enforcing the constraint equations. The hallmark of an unstable code is the (often exponential) growth of of the constraint violations. By periodically solving the constraint equations, constrained evolution reduces the unstable constraint violations before they become too large and crash the simulation. Results of constrained evolution were an improvement in stability (of roughly a factor of two in simulation lifetime) while slowing the evolution of the simulation [1]. For smaller simulations, the elliptic solver *KINSOL* is able to converge to a solution in a reasonable amount of time, however, most of the simulation time is spent solving the constraints rather than evolving the simulation forward in time. For larger simulations ($\sim$ 500 processors), *KINSOL* will not converge to a solution with high accuraccy in 24 hours. Therefore, the use of constrained evolution going

forward is not likely, but may be possible with a faster elliptic solver.

4.2.3 Fisheye Coordinate Transformation

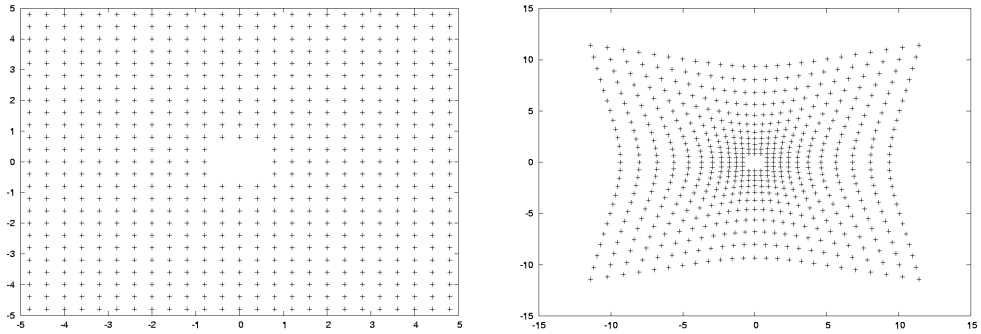
Another cause for instabilites on small computation domains is the propagation of error coming in from the outer boundaries, since the boundary conditions are not constraint preserving. These errors get amplified near the black hole due to the larger gradients in the metric and can cause the code to crash. One way to deal with this problem is to push the outer boundaries further out using a fisheye coordinate transformation.

The transformation is only in the radial coordinate r. By pushing the outer boundaries further away, it takes longer for the errors from the outer boundaries to propagate to the interior of the computation domain. Another nice feature of the fisheye coordinate transformation is that it allows for maintaining high resolution near the center of the computational domain (where the black hole is located) while reducing to lower resolution further from the region of interest. One drawback of the fisheye transformation is coordinate stretching (see Figure 4.1), which can cause instabilities.

Figure 4.3 shows the extended simulation lifetime of a single Kerr-Schild black hole at rest by using the fisheye coordinate transformation

$$\frac{\partial r}{\partial \tilde{r}} = 1 + a + a \tanh k(\tilde{r} - \tilde{r}_c), \tag{4.16}$$

$$r(\tilde{r}) = (1+a)\tilde{r} + \frac{a}{k} \ln \left[\frac{\cosh(k(\tilde{r} - \tilde{r}_c))}{\cosh(k\tilde{r}_c)} \right], \tag{4.17}$$



(a) In fisheye coordinates, the grid remains a Cartesian domain. The domain is -5 to 5 in both $\tilde{x}$ and $\tilde{y}$.

(b) When transforming from fisheye coordinates ($\tilde{r}$) to non-fisheye coordinates (r), the grid will become stretched as above. The domain is -15 to 15 in both x and y.

Figure 4.1: Example of coordinate stretching resulting from a fisheye coordinate transformation.

where r is the non-fisheyed coordinate, $\tilde{r}$ represents the fisheye coordinate, a is a scale factor (not to be confused with the spin of a black hole), $\frac{1}{k}$ gives the width of the transformation range, and $\tilde{r}_c$ marks the center of the transformation. When $\tilde{r} \ll \tilde{r}_c$, $\frac{\partial r}{\partial \tilde{r}} \to 1$. For $\tilde{r} \gg \tilde{r}_c$ the coordinate transformation $\frac{\partial r}{\partial \tilde{r}} \to 1 + 2a$ (see Figure 4.2).

We have since switched to a different fisheye coordinate transformation [27] given by

$$\begin{aligned} r &= a\tilde{r}(1-a)\frac{s}{2\tanh\left(\frac{\tilde{r}_c}{s}\right)} \times \\ &\quad \left[\ln\left(\cosh\frac{\tilde{r}+\tilde{r}_c}{s}\right) - \ln\left(\cosh\frac{\tilde{r}-\tilde{r}_c}{s}\right)\right]. \end{aligned} \tag{4.18}$$

$$\tag{4.19}$$

Here, s is the width of the transformation and $\tilde{r}_c$ is again the center of the

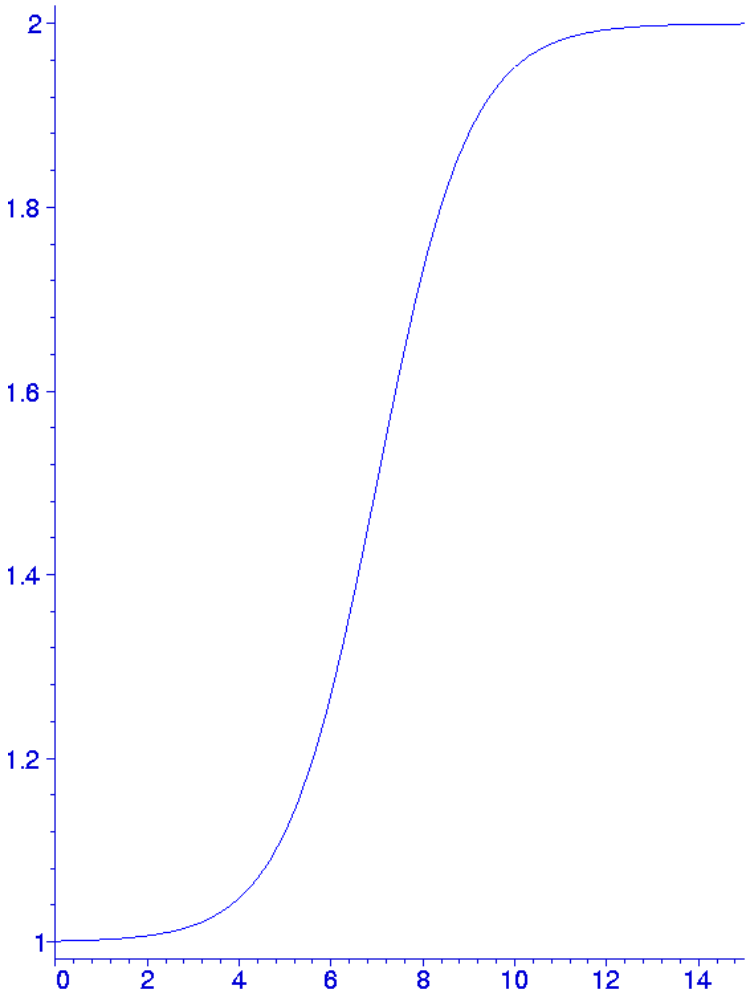


Figure 4.2: $\frac{\partial r}{\partial \tilde{r}}$ versus $\tilde{r}$. $a = .5$, $k = .5$, $\tilde{r}_c = 7$ corresponding to Eq. 4.16.

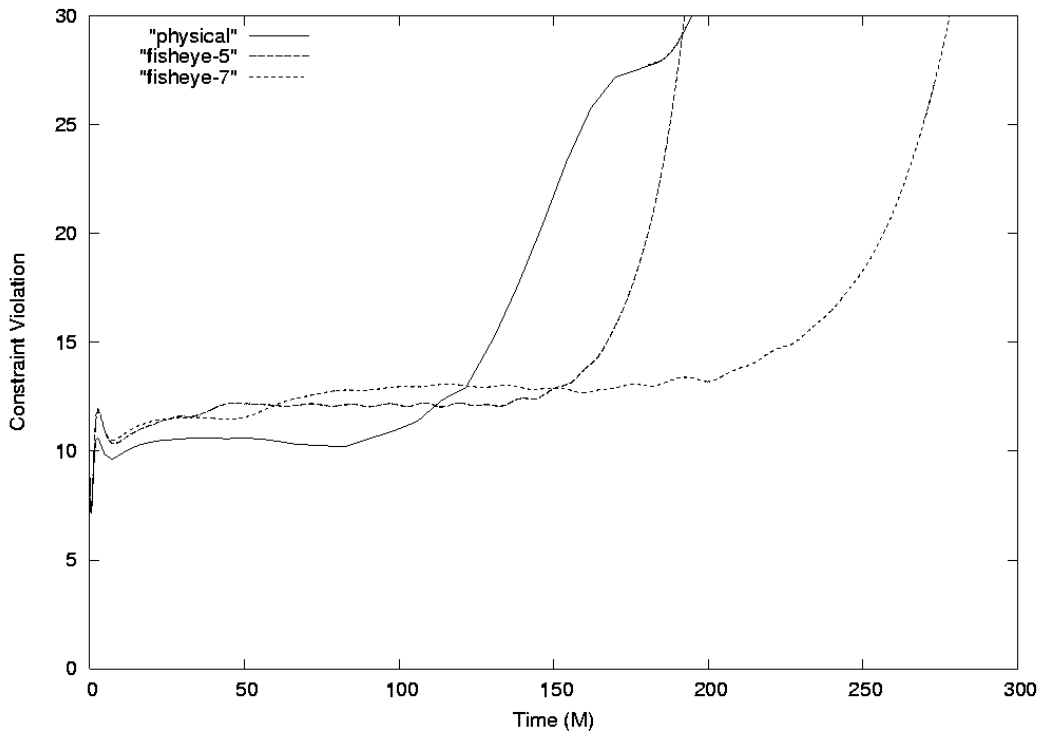


Figure 4.3: Hamiltonian constraint violation versus time. This figure demonstrates the enhanced simulation lifetime from using a fisheye coordinate transformation for a simulation of a single black hole at rest. The solid line is for a simulation with no fisheye transformation, the dashed lines are of simulations using a fisheye transformation. Both fisheye runs used $a = .5$ and $\tilde{r}_c = 7$ for a domain of $[-10M, 10M]$ in each direction with a resolution of $\frac{5 \text{ points}}{M}$, where M is the mass of the black hole. The thicker-dashed run used $k = .5$ while the smaller-dashed run used $k = .7$ (see Eq. 4.16). Clearly, run with $k = .7$ extends the lifetime of the run by factors of approximately two.

transformation. a is still a scale factor, though a slightly different one than in Eq. 4.16. $\frac{\partial r}{\partial \tilde{r}} = 1$ when $\tilde{r} = 0$ and $\frac{\partial r}{\partial \tilde{r}} = a$ for $\tilde{r} \to \infty$. As Figure 4.3 shows, a judiciously selected fisheye transformation can extend the lifetime of the run by factors of approximately two.

Much like having multiple levels of Fixed Mesh Refinement (FMR), it is also possible to have multiple fisheye transformations. For nested fisheye, just add both terms on the right-hand-side of Eq. 4.18 using new variables a_2, s_2, and $\tilde{r}_{c2}$. *openGR* supports up to 10 levels of nested fisheye. While this section is in the chapter on excision, the fisheye transformation is valid regardless of the technique (excision or puncture method).

Chapter 5

Puncture Method

One way of simulating binary black hole mergers that has proven sucessful is to follow the puncture method [15, 16]. We closely follow the method outlined in [17].

5.1 *Puncture:* Initial Data

Using Brill-Lindquist wormhole topology [30], we can construct puncture initial data for any number of black holes. The initial data is conformally flat, thereby the metric can be written as

$$g_{ij} = \psi^4 \tilde{g}_{ij} \tag{5.1}$$

where ψ is a conformal factor and $\tilde{g}_{ij}$ is the conformal metric. Initially, before solving the constraints, the conformal metric is the same as a flat background

$\tilde{g}_{ij} = \delta_{ij}$. We also impose maximal slicing, $K = 0$. The extrinsic curvature can be broken up into

$$K_{ij} = \psi^{-2} A_{ij} + \frac{1}{3} g_{ij} K, \tag{5.2}$$

where K is the trace of the extrinsic curvature and A_{ij} is the trace-free part of the extrinsic curvature. We also impose maximal slicing, $K = 0$. Initial gauge conditions for the lapse and shift consist of $\alpha = 1$ and $\beta^i = 0$. The Hamiltonian constraint is an elliptic equation in terms of the conformal factor,

$$\psi = u + \psi_{BL}. \tag{5.3}$$

with

$$\psi_{BL} = 1 + \sum_{i=1}^{N} \frac{M_i}{2r_i}. \tag{5.4}$$

Thus, the conformal factor is broken up into a singular part (the second term) and an nonsingular term u, which can be solved for. Linear and angular momentum are inserted via the momentum constraints, which allow for Bowen-York solutions [31].

5.2 *Puncture:* BSSN Formulation

When using the puncture method, the initial data is evolved with the strongly hyperbolic BSSN formulation [28, 29]. The BSSN variables are $\tilde{g}_{ij}$, K, ϕ, $\tilde{A}_{ij}$,

and $\tilde{\Gamma}^i$ where

$$\begin{aligned}
\phi &= \ln \psi && (5.5)\\
\tilde{A}_{ij} &= \psi^{-6} A_{ij} && (5.6)\\
\tilde{\Gamma}^i &= -\partial_j \tilde{g}_{ij}. && (5.7)
\end{aligned}$$

The BSSN variables are then evolved via the evolution equations

$$\begin{aligned}
\partial_0 \tilde{g}_{ij} &= -2\alpha \tilde{A}_{ij}, && (5.8)\\
\partial_0 K &= -\nabla^i \nabla_i \alpha + \alpha(\tilde{A}_{ij}\tilde{A}^{ij} + \frac{1}{3}K^2), && (5.9)\\
\partial_0 \phi &= -\frac{1}{6}\alpha K, && (5.10)\\
\partial_0 \tilde{A}_{ij} &= e^{-4\phi}[-\nabla_i \nabla_j \alpha + \alpha R_{ij}]^{TF} + \alpha(K\tilde{A}_{ij} - 2\tilde{A}_{ik}\tilde{A}^k{}_j), && (5.11)\\
\partial_t \tilde{\Gamma}^i &= \tilde{g}^{ij}\partial_j\partial_k\beta^i + \frac{1}{3}\tilde{g}^{ij}\partial_j\partial_k\beta^k + \beta^j\partial_j\tilde{\Gamma}^i - \tilde{\Gamma}^j\partial_j\beta^i + \frac{2}{3}\tilde{\Gamma}^i\partial_j\beta^j \\
&\quad -2\tilde{A}^{ij}\partial_j\alpha + 2\alpha(\tilde{\Gamma}^i{}_{jk}\tilde{A}^{jk} + 6\tilde{A}^{ij}\partial_j\phi - \frac{2}{3}\tilde{g}^{ij}\partial_j K). && (5.12)
\end{aligned}$$

Here, the "TF" represents the trace-free part with respect to the physical metric ($X^{TF}_{ij} = X_{ij} - \frac{1}{3}g_{ij}X^k{}_k$) and $\partial_0 = \partial_t - \pounds_\beta$. The covariant derivative ∇_i is with respect to the physical metric g_{ij}. The Lie derivatives of the non-tensoral quantities are given by [15]

$$\begin{aligned}
\pounds_\beta \phi &= \beta^i\partial_i\phi + \frac{1}{6}\partial_i\beta^i,\\
\pounds_\beta \tilde{g}_{ij} &= \beta^k\partial_k\tilde{g}_{ij} + \tilde{g}_{ik}\partial_j\beta^k + \tilde{g}_{jk}\partial_i\beta^k - \frac{2}{3}\tilde{g}_{ij}\partial_k\beta^k,\\
\pounds_\beta \tilde{A}_{ij} &= \beta^k\partial_k\tilde{A}_{ij} + \tilde{A}_{ik}\partial_j\beta^k + \tilde{A}_{jk}\partial_i\beta^k - \frac{2}{3}\tilde{A}_{ij}\partial_k\beta^k.
\end{aligned}$$

The Ricci tensor $R_{ij} = \tilde{R}_{ij} + R^{\phi}_{ij}$ is given by

$$
\begin{aligned}
R^{\phi}_{ij} &= -2\tilde{\nabla}_i\tilde{\nabla}_j\phi - 2\tilde{g}_{ij}\tilde{\nabla}^k\tilde{\nabla}_k\phi + 4\tilde{\nabla}_i\phi\tilde{\nabla}_j\phi - 4\tilde{g}_{ij}\tilde{\nabla}^k\phi\tilde{\nabla}_k\phi, \quad (5.13)\\
\tilde{R}_{ij} &= -\frac{1}{2}\tilde{g}^{lm}\partial_l\partial_m\tilde{g}_{ij} + \tilde{g}_{k(i}\partial_{j)}\tilde{\Gamma}^k + \tilde{\Gamma}^k\tilde{\Gamma}_{(ij)k} + \\
&\quad \tilde{g}^{lm}(2\tilde{\Gamma}^k{}_{l(i}\tilde{\Gamma}_{j)km} + \tilde{\Gamma}^k{}_{im}\tilde{\Gamma}_{klj}). \quad (5.14)\\
& \quad (5.15)
\end{aligned}
$$

The covariant derivative $\tilde{\nabla}_i$ is with respect to the conformal metric $\tilde{g}_{ij}$.

To better aide stability, it is useful to periodically (at each evolution time step) impose algebraic constraints $det(g) = 1$ and $Tr(A_{ij}) = 0$. Also, we substitute for $\tilde{\Gamma}^i$ everywhere it is undifferentiated with $\tilde{\Gamma}^i = -\partial_j\tilde{g}^{ij}$. When $\tilde{\Gamma}^i$ is differentiated, we leave it as is.

5.3 *Puncture:* Gauge Conditions

To allow the punctures to move, successful gauge conditions have been found to be a covariant form of the "1 + log" slicing [32] conditions for the lapse and a modified $\tilde{\Gamma}$-driver shift condition [16, 33, 34] given by

$$
\begin{aligned}
(\partial_t - \beta^i\partial_i)\alpha &= -2\alpha K, \quad (5.16)\\
(\partial_t - \beta^i\partial_i)\beta^i &= \frac{3}{4}B^i, \quad (5.17)\\
(\partial_t - \beta^i\partial_i)B^i &= (\partial_t - \beta^i\partial_i)\tilde{\Gamma}^i - \eta B^i. \quad (5.18)
\end{aligned}
$$

In our case, $\eta = 1$, but it can also be greater than 1. It has been shown that the BSSN equations with the above gauge choices is strongly hyperbolic and well-posed [35].

Chapter 6

Boundary Conditions

6.1 Implemented Outer Boundary Conditions

openGR supports various outer boundary conditions, such as Dirichlet (exact), Robin, and Sommerfeld. Sommerfeld boundary conditions are currently the outer boundary conditions of choice for our *openGR* simulations [37].

6.1.1 Robin Boundary Conditions

Robin boundary conditions assume that the value of a function falls off radially with some power of r of the form

$$f(r) = f_0 + \frac{k}{r^n}, \tag{6.1}$$

where f_0 is the value of the function as $r \to \infty$, n is the decay rate, and k is a constant. The change as a function of r is given by

$$\frac{\partial f}{\partial r} = -\frac{nk}{r^{n+1}} = -\frac{n}{r}\left(\frac{k}{r^n}\right) = -\frac{n}{r}(f - f_0). \tag{6.2}$$

Robin boundary conditions will work better for the outer boundary of overlapping spherical coordinate patches than they will for a Cartesian coordinate patch. In Cartesian coordinates, we have

$$\frac{\partial f}{\partial x^i} = \frac{\partial f}{\partial r}\frac{\partial r}{\partial x^i} = \frac{\partial f}{\partial r}\frac{x^i}{r}, \tag{6.3}$$

which gives

$$\frac{\partial f}{\partial x^i} = -\frac{nx^i}{r^2}(f - f_0). \tag{6.4}$$

Considering the specific case of the Schwarzschild metric (in Kerr-Schild coordinates) component $g_{xx} = 1 + \frac{2Mx^2}{r^3}$, we can find that it is not always dissipating on the y-z plane moving out in the x direction. We find

$$\begin{aligned} x^2 > \frac{y^2 + z^2}{2} \quad &\to \quad \frac{\partial g_{xx}}{\partial x} < 0 \\ x^2 < \frac{y^2 + z^2}{2} \quad &\to \quad \frac{\partial g_{xx}}{\partial x} > 0. \end{aligned}$$

Thus, the value of g_{xx} is sometimes increasing while going outward in x, making the Robin boundary conditions a poor choice for describing the behavior of the metric in Cartesian coordinates.

Another situation where one must be careful using Robin boundary

conditions is when using a fisheye coordinate transformation. The value of a function will not remain the same as $r \to \infty$, so a value of $f_0 = 1$ for the metric component g_{rr} would no longer be a good choice.

6.1.2 Sommerfeld Radiative Boundary Conditions

Sommerfeld boundary conditions allow for waves propagating outward to leave the spacetime. The conditions on a function f are given by

$$f = f_0 + \frac{u(r - vt)}{r} + h\frac{r + vt}{r}, \tag{6.5}$$

where f_0 is the asymptotic value of f. We see Eq. 6.5 represents an outgoing wave falling off $\sim \frac{1}{r}$. Eq. 6.5 leads to the following equation:

$$\frac{x^i}{r}\frac{df}{dt} + v\frac{df}{dx^i} + \frac{vx^i}{r^2}(f - f_0) = H\frac{vx^i}{r^2}, \tag{6.6}$$

where x^i is normal to the boundaries pointing outward and

$$H = 2\left(\frac{dh(s)}{ds}\right). \tag{6.7}$$

In order to find H, we can expolote H to the boundary assuming that H falls off as a power law:

$$H = \frac{k}{r^n} \tag{6.8}$$

giving

$$d_i H = -n\frac{H}{r}. \tag{6.9}$$

. Empirically, $n = 2$ has been found to be the best value [36].

6.2 *Excision:* Inner Boundary Conditions

While there are no inner boundaries for the puncture method, but when using excision methods, values at the inner boundary must be specified. For the elliptic solve, we use Dirichlet boundary conditions. From the conformal transverse traceless decomposition, the values of the conformal factor and vector potential are given by

$$\phi = 1 \tag{6.10}$$

$$w^i = (0, 0, 0). \tag{6.11}$$

During evolution, the inner boundary is free. Since the inner boundary (excision radius) is inside of the apparent horizon, the values of a variable should be casually disconnected from the rest of the computational domain. Since the inner boundary is free, one-sided finite differencing must be used [1]. Other approaches (Cornell/Caltech) use fully hyperbolic methods and put boundary conditions that outgoing modes exactly vanish at the excision boundary. This leads to stable evolution of excised black holes in those codes.

Chapter 7

Wave Extraction

There are various schemes in use to compute the graviational radiation emitted from a spacetime that are only well-defined in the perturbative regime, such as black hole perturbation theory (Teukolsky [22], Regge-Wheeler [20], and Zerilli [21]). These approaches are all based on knowing the background metric, which is not known for simulations involving strong gravitational fields. For instance, we do not know the mass or spin of the remnant black hole from a binary black hole merger ahead of time. In this chapter, we overview an approach that does not depend on the background metric for extracting gravitational waves. The method that follows will consist of building up the Newman-Penrose formalism which constructs five complex Weyl scalars which contain all the curvature information for a vacuum spacetime. When computing these scalars in a particular class of tetrads, the Quasi-Kinnersly frame, the Weyl scalars will take on precise physical meaning and one Weyl scalar in particular, Ψ_4 will possess the information about the gravitational radiation.

From there, we discuss the method of computing the value of Ψ_4 in terms of scalar quantites. Lastly, we will discuss the mode decompositon of the gravitational radiation and calculation of energy, momentum and angular momentum radiated from the spacetime.

7.1 Tetrad Formalism

It what follows, it will be convenient to work in a non-coordinate basis. A tetrad is a set of four linearly independent basis vectors $\vec{e}_{(a)}$ (where in this chapter the Latin characters inside parentheses are $(0, 1, 2, 3)$), which satisfy the relation

$$e_{(a)\mu} e_{(b)}{}^{\mu} = \eta_{(a)(b)}\,, \tag{7.1}$$

where $\eta_{(a)(b)}$ is a constant independent of the position in spacetime. Metric components in a coordinate frame can be recovered from the tetrads from the relation

$$g_{\mu\nu} = e_{(a)\mu} e^{(a)}{}_{\nu} = \eta^{(a)(b)} e_{(a)\mu} e_{(b)\nu}. \tag{7.2}$$

Notice that the tetrad indices are raised and lowered with $\eta^{(a)(b)}$ and $\eta_{(a)(b)}$ respectively. In the case where the tetrad basis vectors are orthonormal, the metric $\eta_{(a)(b)}$ is Minkowskian. The Schwarzschild metric provides an example:

$$ds^2 = -\left(1 - \frac{2M}{r}\right) dt^2 + \left(1 - \frac{2M}{r}\right)^{-1} dr^2 + r^2 \left(d\theta^2 + sin^2\theta d\phi^2\right). \tag{7.3}$$

A convenient choice for the tetrad is

$$e^{\mu}_{(0)} = \left(1 - \frac{2M}{r}\right)^{\frac{1}{2}} (dt)^{\mu}, \tag{7.4}$$

$$e^{\mu}_{(1)} = \left(1 - \frac{2M}{r}\right)^{-\frac{1}{2}} (dr)^{\mu}, \tag{7.5}$$

$$e^{\mu}_{(2)} = r\,(d\theta)^{\mu}, \tag{7.6}$$

$$e^{\mu}_{(3)} = r\,sin\theta\,(d\phi)^{\mu}. \tag{7.7}$$

7.2 Newman Penrose Formalism

7.2.1 Null Tetrads

The Newman Penrose formalism consists of choosing a particular tetrad where the four basis vectors are null vectors. In particular, two are real (l and k) and two are complex (m and $\bar{m}$), where the bar denotes the complex conjugate. This null tetrad can be constructed from an orthonormal tetrad as follows:

$$l^{\mu} = \frac{1}{\sqrt{2}}\left(e^{\mu}_{(0)} + e^{\mu}_{(1)}\right), \tag{7.8}$$

$$k^{\mu} = \frac{1}{\sqrt{2}}\left(e^{\mu}_{(0)} - e^{\mu}_{(1)}\right), \tag{7.9}$$

$$m^{\mu} = \frac{1}{\sqrt{2}}\left(e^{\mu}_{(2)} + ie^{\mu}_{(3)}\right), \tag{7.10}$$

$$\bar{m}^{\mu} = \frac{1}{\sqrt{2}}\left(e^{\mu}_{(2)} - ie^{\mu}_{(3)}\right). \tag{7.11}$$

The typical choice for $e^{\mu}_{(0)}$ is the unit normal to the spatial hypersurfaces. In spherical coordinates, $e^{\mu}_{(1)}$ is the unit radial vector, and $e^{\mu}_{(2)}$ and $e^{\mu}_{(3)}$ are unit

vectors in angular directions found by performing a Gram-Schmidt orthogonalization procedure. The orthogonality conditions of the null vectors are given by

$$\begin{aligned} l_{\mu}l^{\mu} = k_{\mu}k^{\mu} = m_{\mu}m^{\mu} = \bar{m}_{\mu}\bar{m}^{\mu} = 0, \\ l_{\mu}m^{\mu} = l_{\mu}\bar{m}^{\mu} = k_{\mu}m^{\mu} = k_{\mu}\bar{m}^{\mu} = 0, \\ l_{\mu}k^{\mu} = -m_{\mu}\bar{m}^{\mu} = 1. \end{aligned} \tag{7.12}$$

Using this null tetrad as basis vectors,$\eta_{(\alpha)(\beta)}$ is given by

$$\eta_{(\alpha)(\beta)} = \eta^{(\alpha)(\beta)} = \begin{pmatrix} 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{7.13}$$

The spacetime metric $g_{\mu\nu}$ is

$$g_{\mu\nu} = -l_{\mu}k_{\nu} - k_{\mu}l_{\nu} + m_{\mu}\bar{m}_{\nu} + \bar{m}_{\mu}m_{\nu}. \tag{7.14}$$

7.2.2 Weyl Tensor and Weyl Scalars

The Riemann tensor can be written in terms of the Ricci tensor, Ricci scalar, and an additional traceless term known as the Weyl tensor, which is given by

$$C_{\alpha\beta\mu\nu} = R_{\alpha\beta\mu\nu} - g_{\alpha[\mu}R_{\nu]\beta} + g_{\beta[\mu}R_{\nu]\alpha} + g_{\alpha[\mu}g_{\nu]\beta}R. \tag{7.15}$$

In vacuum, the Ricci tensor and Ricci scalar vanish, leaving the Weyl tensor equal to the Riemann tensor. Thus, the Weyl tensor, which has ten independent components, contains all the of the curvature information of a vacuum spacetime. The ten independent componets of the Weyl tensor can be written in terms of five complex (Weyl) scalars given by

$$\Psi_0 \equiv C_{\alpha\beta\mu\nu}\, l^{\alpha} m^{\beta} l^{\mu} m^{\nu}, \tag{7.16}$$

$$\Psi_1 \equiv C_{\alpha\beta\mu\nu}\, l^{\alpha} n^{\beta} l^{\mu} m^{\nu}, \tag{7.17}$$

$$\Psi_2 \equiv C_{\alpha\eta\mu\nu}\, l^{\mu} m^{\nu} \bar{m}^{\sigma} n^{\nu}, \tag{7.18}$$

$$\Psi_3 \equiv C_{\alpha\beta\mu\nu}\, l^{\alpha} n^{\beta} \bar{m}^{\mu} n^{\nu}, \tag{7.19}$$

$$\Psi_4 \equiv C_{\alpha\beta\mu\nu}\, n^{\alpha} \bar{m}^{\beta} n^{\mu} \bar{m}^{\nu}. \tag{7.20}$$

The Weyl scalars contain all the spacetime curvature information, and being scalars, they do not depend on the choice of coordinates. They do, however, depend on the choice of the tetrad, as we will see in what follows.

7.3 Tetrad transformations

Ultimately, we will determine that Ψ_4 contains the outgoing gravitational wave content in a particular class of frames, known as the Quasi-Kinnersley frame. In getting to that point, we must first discuss the properties of the Weyl scalars under tetrad transformations. There are three types of transformations (Type I, II, and III) which can be performed on a null tetrad. All three are related

to Lorentz transformations.

7.3.1 Type I transformation

A type I transformation leave the null vector $\vec{l}$ unchanged, while rotating the other three null vectors. This rotation is given by

$$l^\mu \rightarrow l^\mu, \tag{7.21a}$$

$$m^\mu \rightarrow m^\mu + al^\mu, \tag{7.21b}$$

$$\bar{m}^\mu \rightarrow \bar{m}^\mu + \bar{a}l^\mu, \tag{7.21c}$$

$$k^\mu \rightarrow k^\mu + \bar{a}m^\mu + a\bar{m}^\mu + a\bar{a}l^\mu. \tag{7.21d}$$

where a is a complex parameter with a complex conjugate $\bar{a}$.

7.3.2 Type II transformation

A type II rotation is given by holding k^μ fixed while rotating the other vectors as given by

$$l^\mu \rightarrow l^\mu + \bar{b}m^\mu + b\bar{m}^\mu + b\bar{b}l^\mu, \tag{7.22a}$$

$$m^\mu \rightarrow m^\mu + bk^\mu, \tag{7.22b}$$

$$\bar{m}^\mu \rightarrow \bar{m}^\mu + \bar{b}k^\mu, \tag{7.22c}$$

$$k^\mu \rightarrow k^\mu, \tag{7.22d}$$

where b is a complex parameter with a complex conjugate $\bar{b}$.

7.3.3 Type III transformation

Type III is a spin/boost transformation, which rescales l^μ and k^μ while performing a null rotation in the complex $m^\mu - \bar{m}^\mu$ plane

$$l^\mu \rightarrow \Lambda^{-1} l^\mu, \tag{7.23}$$

$$m^\mu \rightarrow e^{i\theta} m^\mu, \tag{7.24}$$

$$\bar{m}^\mu \rightarrow e^{-i\theta} \bar{m}^\mu, \tag{7.25}$$

$$k^\mu \rightarrow \Lambda k^\mu, \tag{7.26}$$

where Λ and θ are two real scalars. A *null frame* is defined as a class of null tetrads connected by a spin boost (type III) transformation.

7.4 Curvature Invariants I and J

As will be shown to be useful later, there are scalars that are invarinant to transformations in the tetrad as well as being invariant under coordinate transformations. These curvature invariant quantities I and J are given in terms of the Weyl tensor by

$$\mathrm{I} = \frac{1}{16}\Big(C_{\mu\nu}{}^{\rho\sigma} C_{\rho\sigma}{}^{\mu\nu} - i\, C_{\mu\nu}{}^{\rho\sigma}\, {}^{\sim}C_{\rho\sigma}{}^{\mu\nu} \Big) \tag{7.27}$$

and

$$\mathrm{J} = \frac{1}{96}\left(C_{\mu\nu}{}^{\rho\sigma} C_{\rho\sigma}{}^{\alpha\beta} C_{\alpha\beta}{}^{\mu\nu} - C_{\mu\nu}{}^{\rho\sigma} C_{\rho\sigma}{}^{\alpha\beta}\, {}^{\sim}C_{\alpha\beta}{}^{\mu\nu} \right), \tag{7.28}$$

where

$$^{\sim}C_{\mu\nu}{}^{\rho\sigma} \equiv \frac{1}{2}{}^{4}\varepsilon_{\mu\nu}{}^{\delta\lambda}C_{\delta\lambda\sigma\rho} \tag{7.29}$$

is the Hodge dual of the Weyl tensor. Expressed in terms of the Weyl scalars, I and J take the form:

$$\mathrm{I} = \Psi_4\Psi_0 - 4\Psi_3\Psi_1 + 3{\Psi_2}^2, \tag{7.30}$$

$$\mathrm{J} = det\begin{vmatrix} \Psi_4 & \Psi_3 & \Psi_2 \\ \Psi_3 & \Psi_2 & \Psi_1 \\ \Psi_2 & \Psi_1 & \Psi_0 \end{vmatrix}. \tag{7.31}$$

7.5 Principal Null Directions

When light from a distance source is bent from passing by massive objects, the resulting image will be distorted. A measure of the distortion D is given by

$$D = \frac{1}{2}\,C\,(r,\theta,\phi)\,r^2. \tag{7.32}$$

where $C(r,\theta,\phi)$ is the projection of the Weyl tensor on the tangent plane of the celestial sphere. There are four directions where $C(r,\theta,\phi) = 0$, known as the principal null directions. Penrose found these four principal null directions by setting $\Psi_4 = 0$ after a type I rotation, i.e.

$$\Psi_4^{\,I} = \Psi_4 + 4\bar{a}\Psi_3 + 6\bar{a}^2\Psi_2 + 4\bar{a}^3\Psi_1 + \bar{a}^4\Psi_0 = 0. \tag{7.33}$$

The four solutions for $\bar{a}$ in Eq. 7.33 can be obtained by first introducing a new variable

$$y = \Psi_4\,\bar{a} + \Psi_3. \tag{7.34}$$

In terms of y, Eq. 7.33 is now

$$y^4 + 6\,H\,y^2 + 4\,G\,y + K = 0, \tag{7.35}$$

where

$$\begin{aligned} H &= \Psi_0\,\Psi_2 - \Psi_1^2, && \text{(7.36a)} \\ G &= \Psi_0^2\,\Psi_3 - 3\,\Psi_0\,\Psi_1\,\Psi_2 + 2\,\Psi_1^3, && \text{(7.36b)} \\ K &= \Psi_0^2\,\mathrm{I} - 3\,H^2. && \text{(7.36c)} \end{aligned}$$

Since Eq. 7.35 lacks a third power of y, its solutions satisfy

$$y_1 + y_2 + y_3 + y_4 = 0. \tag{7.37}$$

We replace these solutions with three other variables (α, β, γ) as follows:

$$\begin{aligned} y_1 &= \frac{1}{2}\Big(\alpha + \beta + \gamma\Big), && \text{(7.38a)} \\ y_2 &= \frac{1}{2}\Big(\alpha - \beta - \gamma\Big), && \text{(7.38b)} \\ y_3 &= \frac{1}{2}\Big(-\alpha + \beta - \gamma\Big), && \text{(7.38c)} \\ y_4 &= \frac{1}{2}\Big(-\alpha - \beta + \gamma\Big). && \text{(7.38d)} \end{aligned}$$

Here, it is straightforward to see the sum does vanish. By substituting the above in to Eq. 7.33, we get the relation

$$\begin{aligned} \alpha^2 + \beta^2 + \gamma^2 &= -12\,H, && (7.39a)\\ \alpha\,\beta\,\gamma &= 4\,G, && (7.39b)\\ \alpha^2\beta^2 + \alpha^2\gamma^2 + \beta^2\gamma^2 &= 36\,H^2 - 4\,K. && (7.39c) \end{aligned}$$

From this, we can again introduce three new variables

$$\begin{aligned} \lambda_1 &= \frac{\alpha^2 + 4\,H}{2\,\Psi_4}, && (7.40a)\\ \lambda_2 &= \frac{\beta^2 + 4\,H}{2\,\Psi_4}, && (7.40b)\\ \lambda_3 &= \frac{\gamma^2 + 4\,H}{2\,\Psi_4}. && (7.40c) \end{aligned}$$

which must satisfy the following conditions

$$\begin{aligned} \lambda_1 + \lambda_2 + \lambda_3 &= 0, && (7.41a)\\ \lambda_1\,\lambda_2 + \lambda_1\,\lambda_3 + \lambda_2\,\lambda_3 &= 48\,H^2 + 8\,H\left(\alpha^2 + \beta^2 + \gamma^2\right)\\ &+ \left(\alpha^2\,\beta^2 + \alpha^2\,\gamma^2 + \beta^2\,\gamma^2\right), && (7.41b)\\ \lambda_1\,\lambda_2\,\lambda_3 &= 64\,H^3 + 16\,H^2\left(\alpha^2 + \beta^2 + \gamma^2\right)\\ &+ 4\,H\left(\alpha^2\,\beta^2 + \alpha^2\,\gamma^2 + \beta^2\,\gamma^2\right) + \alpha^2\,\beta^2\,\gamma^2. && (7.41c) \end{aligned}$$

This gives a third order polynomial

$$\lambda^3 - I\lambda + 2\,J = 0, \tag{7.42}$$

which we can be solved for solutions of the form

$$\lambda_1 = - \left(P + \frac{\mathrm{I}}{3P}\right), \tag{7.43}$$

$$\lambda_2 = - \left(e^{\frac{2\pi i}{3}} P + e^{\frac{4\pi i}{3}} \frac{\mathrm{I}}{3P}\right), \tag{7.44}$$

$$\lambda_3 = - \left(e^{\frac{4\pi i}{3}} P + e^{\frac{2\pi i}{3}} \frac{\mathrm{I}}{3P}\right), \tag{7.45}$$

where

$$P = \left[\mathrm{J} + \sqrt{\mathrm{J}^2 - \left(\frac{\mathrm{I}}{3}\right)^3}\right]^{\frac{1}{3}}. \tag{7.46}$$

Comparing Eqs. 7.40 and 7.43 gives a final solution for the principal null directions in terms of α, β, and γ:

$$\bar{a}_1 = -\frac{\Psi_3 + \frac{1}{2}\Big(\alpha + \beta + \gamma\Big)}{\Psi_4}, \tag{7.47a}$$

$$\bar{a}_2 = -\frac{\Psi_3 + \frac{1}{2}\Big(\alpha - \beta - \gamma\Big)}{\Psi_4}, \tag{7.47b}$$

$$\bar{a}_3 = -\frac{\Psi_3 + \frac{1}{2}\Big(-\alpha + \beta - \gamma\Big)}{\Psi_4}, \tag{7.47c}$$

$$\bar{a}_4 = -\frac{\Psi_3 + \frac{1}{2}\Big(-\alpha - \beta + \gamma\Big)}{\Psi_4}. \tag{7.47d}$$

$$\tag{7.47e}$$

7.5.1 Petrov Classification

Petrov Classification	
Petrov Type	
I	Four distinct principal null directions
II	Four distinct principal null directions
D	Four distinct principal null directions
III	Three distinct principal null directions coincide
N	All four principal null directions coincide

Table 7.1: The different Pretrov spacetimes are determined by how the principal null directions coincide.

The various Petrov classifications are listed in Table 7.5.1. The different classifications are based on the how the principal null directions coincide. Of particular interest is the Petrov type D and type I spacetimes.

- **Petrov Type D:** A Petrov type D spacetime has two pairs of coinciding principal null directions. Petrov type D spacetimes have the special relation that $27J^2 = I^3$. If the null vector $\vec{l}$ is pointed along one set of repeated principal null directions, this sets $\Psi_4 = \Psi_3 = 0$. Setting the null vector $\vec{k}$ along the other couple of repeated principal null directions, we find that $\Psi_0 = \Psi_1 = 0$. Only Ψ_2 is nonvanishing. It can be shown that Ψ_2 contains all the background infromation. In order to test the computations of the Weyl scalars, we note that both Schwarzschild and Kerr spacetimes are of Petrov type D. For a Schwarzschild spacetime we find

$$(\Psi_2)_{Schwarschild} = -\frac{M}{r^3}. \tag{7.48}$$

For a Kerr spacetime

$$(\Psi_2)_{Kerr} = -\frac{M}{(r+ia)^3}, \tag{7.49}$$

where a is the spin parameter. A type D frame is also known as a Kinnersley frame.

- **Petrov Type I:** For a Petrov type I spacetime, none of the principal null directions coincide. By performing tetrad transformations, one is able to set two of the Weyl scalars to zero. We will take advantage of this and in practice will set $\Psi_1 = \Psi_3 = 0$, a condition necessary for finding transverse frames which we will discuss shortly.

7.6 Quasi-Kinnersley Frame

Perturbation theory has shown that if the choice of a tetrad is infinitesimally close to a type D tetrad, the Weyl scalars Ψ_0 and Ψ_4 are found to be invariant under tetrad and gauge invariant, giving them precise physicsal meaning (in a more general spacetime, information about the spacetime is mixed between the Weyl scalars). Ψ_4 is the outgoing radiation while Ψ_0 is ingoing radiation. The problem is that for numerical simulations, the parameters (mass, spin) of the remnant black hole (produced from the merger of two black holes) are not known *a priori*. In this chapter, we will show the method in which to choose the correct tetrad frame based on the knowledge that the spacetime is type D (and thus has two pairs of coinciding principal null directions).

- For type I spacetimes, Ψ_1 an Ψ_3 represent gauge choices and can be set to zero. Transverse frames are have the condition that $\Psi_1 = \Psi_3 = 0$.

- A Kinnersley frame for a type D spacetime is a frame where the two real null tetrad vectors ($\vec{l}$ and $\vec{k}$ coincide with the two repeated null directions of the Weyl tensor.

- A quasi-Kinnersley frame of a type I spacetime is the frame that converges to the Kinnersley frame when $\mathcal{S} = \frac{27J^2}{I^3} \rightarrow 1$. The quasi-Kinnersly frame is a transverse frame with the extra condition is that as the spacetime converges to a Kinnersley frame, the Weyl scalars Ψ_0 and Ψ_4 must also tend towards zero. We introduce a radiation scalar $\xi = \Psi_0\Psi_4$ which tends towards zero in a quasi-Kinnersley frame as $\mathcal{S} \rightarrow 1$.

In a Petrov I spacetime, the quasi-Kinnersley frame can be found by performing a type I rotation followed by a type II rotation:

$$
\begin{aligned}
\Psi_0 &\rightarrow \Psi_0^I = \Psi_0, \\
\Psi_1 &\rightarrow \Psi_1^I = \Psi_1 + \bar{a}\Psi_0, \\
\Psi_2 &\rightarrow \Psi_2^I = \Psi_2 + 2\bar{a}_1^{\Psi} + \bar{a}^2\Psi_0, \\
\Psi_3 &\rightarrow \Psi_3^I = \Psi_3 + 3\bar{a}\Psi_2 + 3\bar{a}^2\Psi_1 + \bar{a}^3\Psi_0, \\
\Psi_4 &\rightarrow \Psi_4^I = \Psi_4 + 4\bar{a}\Psi_3 + 6\bar{a}^2\Psi_2 + 4\bar{a}^3\Psi_1 + \bar{a}^4\Psi_0, \\
\Psi_0^I &\rightarrow \Psi_0^{II} = \Psi_0^I + 4b\Psi_1^I + 6b^2\Psi_2^I + 4b^3\Psi_3^I + b^4\Psi_4^I, \\
\Psi_1^I &\rightarrow \Psi_1^{II} = \Psi_1^I + 3b\Psi_2^I + 3b^2\Psi_3^I + b^3\Psi_4^I, \\
\Psi_2^I &\rightarrow \Psi_2^{II} = \Psi_2^I + 2b\Psi_3^I + b^2\Psi_4^I, \\
\Psi_3^I &\rightarrow \Psi_3^{II} = \Psi_3^I + b\Psi_4^I, \\
\Psi_4^I &\rightarrow \Psi_4^{II} = \Psi_4^I.
\end{aligned}
\tag{7.50}
$$

Plugging $\Psi_1^{II} = \Psi_3^{II} = 0$ into the rotation equations, we are left to solve the following

$$
\begin{gathered}
\Psi_3 + 3\bar{a}\Psi_2 + 3\bar{a}^2\Psi_1 + \bar{a}^2\Psi_0 + \\
b\left(\Psi_4 + 4\bar{a}\Psi_3 + 6\bar{a}^2\Psi_2 + \bar{a}^3\Psi_1 + \bar{a}^4\Psi_0\right) = 0,
\end{gathered}
\tag{7.51a}
$$

$$
\begin{gathered}
\Psi_1 + \bar{a}\Psi_0 + 3b\left(\Psi_2 + \bar{a}\Psi_1 + \bar{a}^2\Psi_0\right) + \\
3b^2\left(\Psi_3 + 3\bar{a}\Psi_2 + 3\bar{a}^2\Psi_1 + \bar{a}^3\Psi_0\right) + \\
b^3\left(\Psi_4 + 4\bar{a}\Psi_3 + 6\bar{a}^2\Psi_2 + 4\bar{a}^3\Psi_1 + \bar{a}^4\Psi_0\right) = 0.
\end{gathered}
\tag{7.51b}
$$

For a Petrov type I spacetime b is given by

$$b = -\frac{\Psi_3 + 3\bar{a}\Psi_2 + 3\bar{a}^2\Psi_1 + \bar{a}^2\Psi_0}{\Psi_4 + 4\bar{a}\Psi_3 + 6\bar{a}^2\Psi_2 + \bar{a}^3\Psi_1 + \bar{a}^4\Psi_0}. \tag{7.52}$$

Substituting this into (7.51b) gives a sixth order polynomial for $\bar{a}$

$$\mathcal{P}_6\bar{a}^6 + \mathcal{P}_5\bar{a}^5 + \mathcal{P}_4\bar{a}^4 + \mathcal{P}_3\bar{a}^3 + \mathcal{P}_2\bar{a}^2 + \mathcal{P}_1\bar{a}^1 + \mathcal{P}_0 = 0. \tag{7.53}$$

The parameters $\mathcal{P}_n$ are given by

$$\begin{aligned}
\mathcal{P}_6 &= -\Psi_3\Psi_0^2 - 2\Psi_1^3 + 3\Psi_2\Psi_1\Psi_0, && (7.54)\\
\mathcal{P}_5 &= -2\Psi_3\Psi_1\Psi_0 - \Psi_0^2\Psi_4 + 9\Psi_2^2\Psi_0 - 6\Psi_2\Psi_1^2, && (7.55)\\
\mathcal{P}_4 &= -5\Psi_1\Psi_4\Psi_0 - 10\Psi_3\Psi_1^2 + 15\Psi_3\Psi_2\Psi_0, && (7.56)\\
\mathcal{P}_3 &= -10\Psi_4\Psi_1^2 + 10\Psi_3^2\Psi_0, && (7.57)\\
\mathcal{P}_2 &= 5\Psi_3\Psi_0\Psi_4 + 10\Psi_1\Psi_3^2 - 15\Psi_1\Psi_2\Psi_4, && (7.58)\\
\mathcal{P}_1 &= 2\Psi_3\Psi_1\Psi_4 + \Psi_4^2\Psi_0 - 9\Psi_2^2\Psi_4 + 6\Psi_2\Psi_3^2, && (7.59)\\
\mathcal{P}_0 &= \Psi_1\Psi_4^2 + 2\Psi_3^3 - 3\Psi_2\Psi_3\Psi_4. && (7.60)
\end{aligned}$$

Given $\bar{a}$, b can be found. Even though eq.(7.53) is sixth order, there are only 3 transverse frames because of the degeneracy in $\vec{l}$ and $\vec{k}$. An exchange of $\vec{l}$

and $\vec{k}$

$$\begin{aligned} \Psi_0 &\leftrightarrow \bar{\Psi}_4, \\ \Psi_2 &\leftrightarrow \Psi_2, \\ \Psi_4 &\leftrightarrow \bar{\Psi}_0, \end{aligned} \tag{7.61}$$

Thus we have found that there is three transverse frames for a Petrov type I spacetime, and we will find that one of them is the quasi-Kinnersley frame.

7.7 Finding the quasi-Kinnersley frame

In terms of the parameters α, β, and γ, the solutions for the three transverse frames are given by

$$\bar{a}_{\pm}^{I} = \frac{1}{2\,\alpha}\left[\beta\,\gamma\,\sqrt{(\alpha^2-\beta^2)\,(\alpha^2-\gamma^2)}\right], \tag{7.62a}$$

$$\bar{a}_{\pm}^{II} = \frac{1}{2\,\beta}\left[\alpha\,\gamma\,\sqrt{(\beta^2-\alpha^2)\,(\beta^2-\gamma^2)}\right], \tag{7.62b}$$

$$\bar{a}_{\pm}^{III} = \frac{1}{2\,\gamma}\left[\alpha\,\beta\,\sqrt{(\gamma^2-\alpha^2)\,(\gamma^2-\beta^2)}\right]. \tag{7.62c}$$

The $\pm$ is due to the degeneracy in $\vec{l}$ and $\vec{k}$. To find the quasi-Kinnersley frame, we need $S \to 1 \implies \xi \to 0$. It can be seen that $S \to 1 \implies P \to \sqrt[3]{J} \implies$

${}^3\sqrt{J} \to \sqrt{I}$, Using eq.(**??**), the following behavior of λ is found

$$\lambda_1 \rightarrow -2\sqrt{\frac{\mathrm{I}}{3}}, \tag{7.63a}$$

$$\lambda_2 \rightarrow \sqrt{\frac{\mathrm{I}}{3}}, \tag{7.63b}$$

$$\lambda_3 \rightarrow \sqrt{\frac{\mathrm{I}}{3}}. \tag{7.63c}$$

Thus Ψ_2 in the three frames take the value

$$\Psi_2^{\,I} = \frac{\lambda_1}{2}, \tag{7.64a}$$

$$\Psi_2^{\,II} = \frac{\lambda_2}{2}, \tag{7.64b}$$

$$\Psi_2^{\,III} = \frac{\lambda_3}{2}, \tag{7.64c}$$

which gives for the radiation scalar ξ

$$\xi^{\,I} = \frac{(\lambda_2 - \lambda_3)^2}{4}, \tag{7.65a}$$

$$\xi^{\,II} = \frac{(\lambda_1 - \lambda_3)^2}{4}, \tag{7.65b}$$

$$\xi^{\,III} = \frac{(\lambda_1 - \lambda_2)^2}{4}. \tag{7.65c}$$

(7.63a) show that the equations for ξ in the case $S \to 1$ are

$$\xi^{I} \rightarrow 0, \tag{7.66a}$$

$$\xi^{II} \rightarrow \frac{3\,\mathrm{I}}{4}, \tag{7.66b}$$

$$\xi^{III} \rightarrow \frac{3\,\mathrm{I}}{4}. \tag{7.66c}$$

Thus, the transverse frame I is the quasi-Kinnersley frame.

7.8 Ψ_4 in Terms of Scalar Quantities

Following the treatment by Nerozzi [9], we will now describe the treatment for computing Ψ_4 in *openGR*. It should be noted that implementing his most recent work [8] is the subject of future work. From the Weyl tensor, we can construct the electric and magnetic parts of the Weyl tensor as follows:

$$E_{\alpha\gamma} = -C_{\alpha\beta\gamma\delta} e^{\mu}_{(0)} e^{\nu}_{(0)}, \tag{7.67}$$

$$B_{\alpha\gamma} = -\frac{1}{2}\varepsilon^{\mu\nu}_{\alpha\beta} C_{\alpha\beta\mu\nu} e^{\mu}_{(0)} e^{\nu}_{(0)}, \tag{7.68}$$

where we recall $e_{(0)^{\mu}}$ is the timelike vector (taken as the unit vector normal to the hypersurface in simulations) and $\varepsilon^{\mu\nu}_{\alpha\beta}$ is the four dimensional Levi-Civita tensor. We can then express the Weyl tensors in terms of the electric and

magnetic parts of the Weyl tensor, given by

$$\Psi_0 = -(E_{\beta\gamma} - iB_{\beta\gamma})m^\beta m^\gamma, \tag{7.69a}$$

$$\Psi_1 = -(E_{\beta\gamma} - iB_{\beta\gamma})m^\beta e^\gamma_{(1)}, \tag{7.69b}$$

$$\Psi_2 = -(E_{\beta\gamma} - iB_{\beta\gamma})e^\beta_{(1)} e^\gamma_{(1)}, \tag{7.69c}$$

$$\Psi_3 = -(E_{\beta\gamma} - iB_{\beta\gamma})\bar{m}^\beta e^\gamma_{(1)}, \tag{7.69d}$$

$$\Psi_4 = -(E_{\beta\gamma} - iB_{\beta\gamma})\bar{m}^\beta \bar{m}^\gamma, \tag{7.69e}$$

Determining the transverse frame where $\Psi_1 = \Psi_3 = 0$ and calculating the eigenvalues of the electric and magnetic parts of the Weyl tensor, the Weyl scalar Ψ_4 can be expressed in terms of a modulus and phase given by

$$\mathrm{Re}(\Psi_4)_{TF} = -\sqrt{3}|\mathcal{E}| \sin\left(\Theta_\mathcal{E} + \frac{2k\pi}{3}\right), \tag{7.70}$$

$$\mathrm{Im}(\Psi_4)_{TF} = \sqrt{3}|\mathcal{B}| \sin\left(\Theta_\mathcal{B} + \frac{2k\pi}{3}\right), \tag{7.71}$$

where "TF" denotes being in the transverse frame. In *openGR*, $k = 0$, but in general it can assume values $\{-1, 0, 1\}$. The moduli $\mathcal{E}$ and $\mathcal{B}$ as well as the

phases $\Theta_{\mathcal{E}}$ and $\Theta_{\mathcal{B}}$ are given by

$$|\mathcal{E}| = \sqrt{\frac{E_{\alpha\beta}E^{\alpha\beta}}{6}}, \tag{7.72a}$$

$$\Theta_{\mathcal{E}} = \frac{1}{3}\arccos\left[\sqrt{6}\left(\frac{E_{\alpha\beta}E^{\beta}{}_{\gamma}E^{\gamma\alpha}}{[E_{\alpha\beta}]^{\frac{3}{2}}}\right)\right], \tag{7.72b}$$

$$|\mathcal{B}| = \sqrt{\frac{B_{\alpha\beta}B^{\alpha\beta}}{6}}, \tag{7.72c}$$

$$\Theta_{\mathcal{B}} = \frac{1}{3}\arccos\left[\sqrt{6}\left(\frac{B_{\alpha\beta}B^{\beta}{}_{\gamma}B^{\gamma\alpha}}{[B_{\alpha\beta}]^{\frac{3}{2}}}\right)\right], \tag{7.72d}$$

Given the summations above, the Weyl scalar Ψ_4 is given in terms of scalar quantities.

7.9 Mode Decomposition of Ψ_4

Projecting Ψ_4 on the the spherical harmonics of spin weight $s = -2$ allows us to compute the contributions of the individual l, m modes. The scalar product of Ψ_4 and Y^{-2}_{lm} gives

$$A_{lm} = \langle Y^{-2}_{lm}, \Psi_4 \rangle = \int_0^{2\pi}\int_0^{\pi} \Psi_4 \bar{Y}^{-2}_{lm} \sin\theta d\theta d\phi. \tag{7.73}$$

where the spin-weighted spherical harmonics Y^s_{lm} are given by

$$Y^s_{lm}(\theta,\phi) = (-1)^s\sqrt{\frac{2l+1}{4\pi}}d^l_{m(-s)}(\theta)e^{im\phi}, \tag{7.74}$$

where $d^l_{ms}(\theta)$ are the Wigner d-functions given by

$$d^l_{ms}(\theta) = \sum_{t=C_1}^{C_2} \frac{(-1)^t[(l+m)!(l-m)!(l+s)!(l-s)!]^{\frac{1}{2}}}{(l+m-t)!(l-s-t)!t!(t+s-m)!} \left(\frac{\cos\theta}{2}\right)^{2l+m-s-2t} \left(\frac{\sin\theta}{2}\right)^{2t+s-m}, \quad (7.75)$$

where $C_1 = \max(0, m-s)$ and $C_2 = min(l+m, l-s)$.

7.10 Mass, Momentum and Angular Momentum

As gravitational waves propagate away from a source, they carry away energy and momentum. For a system, we can compute the rates at which energy and momenta are radiated in terms of the Weyl scalar Ψ_4. These are given by

$$\frac{dE}{dt} = \lim_{r\to\infty}\left[\frac{r^2}{16\pi}\int_\Omega \left|\int_{-\infty}^{t}\Psi_4 dt\right|^2 d\Omega\right], \quad (7.76)$$

$$\frac{dP_i}{dt} = -\lim_{r\to\infty}\left[\frac{r^2}{16\pi}\int_\Omega l_i\left|\int_{-\infty}^{t}\Psi_4 dt\right|^2 d\Omega\right], \quad (7.77)$$

$$\frac{dJ_z}{dt} = -\lim_{r\to\infty}\left\{\frac{r^2}{16\pi}\mathrm{Re}\left[\int_\Omega\left(\partial_\phi\int_{-\infty}^{t}\Psi_4 dt\right)\left(\int_{-\infty}^{t}\int_{-\infty}^{t'}\bar{\Psi}_4 dt dt'\right)d\Omega\right]\right\}, \quad (7.78)$$

where $l_i = (-\sin\theta\cos\phi, -\sin\theta\sin\phi, -\cos\phi)$. The expression for the radiated energy simplifies when we expand Ψ_4 in the modes l, m, enables us to calculate

the energy radiated in individual modes

$$\frac{dE}{dt} = \lim_{r\to\infty}\left[\frac{r^2}{16\pi}\left|\int_{-\infty}^{t}\sum_{l,m}A_{l,m}dt\right|^2\right] \tag{7.79}$$

In typical simulations, most of the energy is radiated in the $l = 2, m = \pm 2$ modes.

The total amount of energy in a spacetime is given by the ADM Energy M_{ADM}, which is also known as the ADM Mass [17]. Given that our numerical simulations do not extend out to spacial infinity, the energy in the spacetime is computed within a given sphere S_r of radius r, given by

$$E(r) = \frac{1}{16\pi}\int_{S_r}\sqrt{g}g^{ij}g^{kl}(g_{ik,j} - g_{ij,k})dS_l \tag{7.80}$$

The ADM Energy is given by taking the limit as r goes to infinity

$$M_{ADM} = \lim_{r\to\infty} E(r).$$

The total amount of linear momentum P_i and angular momentum J_i are computed from

$$P_i(r) = \frac{1}{8\pi}\int_{S_r}\sqrt{g}(K^j{}_i - \delta^j{}_i K)dS_j \tag{7.81}$$

$$J_i(r) = \frac{1}{8\pi}\varepsilon_{ij}{}^k\int_{S_r}\sqrt{g}x^j(K^m{}_k - \delta^m{}_k K)dS_m \tag{7.82}$$

The total linear momentum P_i and angular momentum J_i are given by

$$\begin{aligned} P_i &= \lim_{r\to\infty} P_i(r), \\ J_i &= \lim_{r\to\infty} J_i(r). \end{aligned}$$

The three dimensional Levi-Civita tensor is a contraction of the four dimensional Levi-Civita tensor

$$\varepsilon^{ijk} \equiv {}^4\varepsilon^{\alpha\beta\gamma\delta}\,\hat{n}_\delta \tag{7.83}$$

$$= \hat{x}^i \cdot \left(\hat{x}^j \times \hat{x}^k\right) = \sqrt{g}\,[1\,2\,3]^{ijk}, \tag{7.84}$$

where

$$[123]^{ijk} = \begin{cases} +1 & \text{for } (i,j,k) \in \{(1,\,2,\,3)\,,\,(3,\,1,\,2)\,,\,(2,\,3,\,1)\}\,, \\ -1 & \text{for } (i,j,k) \in \{(1,\,3,\,2)\,,\,(2,\,1,\,3)\,,\,(3,\,2,\,1)\}\,, \\ 0 & \text{for any other combination.} \end{cases} \tag{7.85}$$

Currently in *openGR*, wave extraction is a work in progress. The computation of the Weyl scalars in terms of scalars (consisting of the electric and magnetic part of the Weyl scalar) is implemented. Courtesy of Uli Spherhake, we also have scripts to decompose Ψ_4 in to l, m modes. However, we have not yet implement the mode decomposition scripts into *openGR*. This should be remedied in short order.

Chapter 8

Scaling and Performance

We now turn our attention to matters of performance. In this chapter, we discuss how *openGR* scales on ever-increasing numbers of processors, taking a look at both strong and weak scaling. Sec. 8.5 will show the results of a head-on collision simulation that was allowed to run for 24 hours. Sec. 8.7 investigates how *openGR* performs in terms of memory.

8.1 Scaling

A relation between the wall clock time t and the number of processors n can be written as

$$t = an^b. \tag{8.1}$$

Taking the log of both sides gives

$$\log t = \log a + b \log n.$$

The $\log a$ term is a constant and b represents the slope on a log-log graph. Using Eq. 8.1 when comparing two simulations gives

$$\frac{t_2}{t_1} = \left(\frac{n_2}{n_1}\right)^b, \tag{8.2}$$

providing the ratio of wall clock times for varying number of processors.

Strong scaling consists on measuring performance by running the same simulation on a varying number of processors. In the ideal case, doubling the number of processors should decrease the processor run time (wall clock time) by a factor of two. On a log-log graph, this would correspond to a slope b of -1, giving $t \sim \frac{1}{n}$. Such an ideal case is not typically realized due to the overhead from communication between processors. As the number of processors carrying out a job increases, the communication overhead per processor remains the same while the portion of the job that each processor carries out is decreased. Eventually, a limit will be reached where the communication overhead per processor (ghostzone cells) will become comparable to the size of the job per processor (non-ghostzone cells). At this point, there is no further appreciable speed-up of the simulation when running on more processors, corresponding to a leveling off in the wall clock time.

Weak scaling consists of varying the overall size of the simulation such that each processor is running the same size of job. For instance, consider a test case (Job A) run on some number of processors. Doubling the resolution of the simulation in each spatial direction means increases the size of the job by a factor of eight (Job B). Weak scaling is measured by comparing the wall

clock times of Job A to Job B, when Job B is run on eight times the number of processors. Ideally, both jobs should be carried out in the same amount of time. Deviations away from the ideal case place a limit on the size of the job that can be run.

Since the majority of wall clock time of a simulation is spent evolving a given spacetime, we present scaling results of the wall clock time spent in PVODE (the evolution portion of *openGR*). All of these examples were run o the TACC computer *Ranger*.

8.2 Single Puncture Scaling on Unigrid Domain

Figure 8.1 shows the strong scaling of simulations of a single puncture at rest in a unigrid domain. Each simulation was carried out to a time $t = 4.5M$, where M is the mass of the puncture. Details of the simulations are shown in Table 8.1. Over the range of 16 to 256 processors, the slope of curve B is $-.95$. The ideal situation is to have a slope of -1, which would correspond to doing the same job in half the time if using twice the number of processors. Over the range of 32 to 1024 processors, the slope of curve C is $-.89$. The range of 512 to 4096 processors on curve D is $-.85$. On the curves, some points are outliers, consisting of simulations on 192, 768, and 1536 processors. These differences are due to the load balancing of the processor layout and the outliers plot a similar slope (curve D in Figure 8.1).

Figure 8.2 shows the weak scaling of the simulations of a single puncture at rest (see Table 8.1). In the ideal case, the slope of the curves would be zero, which would correspond to being able to do a job twice the size on twice the number of processors in the same amount of time. In our case, the average slope for all the curves combined is .23. For simulations of 1536 processors or less, the slope is .09. We conclude that unigrid codes run with good scaling efficiency at lest up to $4,096$ processors.

Evolved to $t = 4.5M$					
Job	A	B	C	D	E
Resolution	$5M$	$2.5M$	$1.25M$	$\frac{5M}{8} = .625M$	$\frac{5M}{16} = .3125M$
# of Points	40^3	80^3	160^3	320^3	640^3
Strong Scaling:					
Proc Range		$16 - 256$	$32 - 1024$	$512 - 4096$	
Slope		$-.95$	$-.89$	$-.85$	
Weak Scaling:					
Average Slope	.23				

Table 8.1: Simulations of a single puncture at rest without mesh refinment (unigrid) corresponding to the plots in Figure 8.4. The physical domain is $-100M$ to $100M$ in each spatial direction, where M is the mass of the puncture. Note that each subsequent job is a factor of two finer in resolution, and thus a factor of eight larger in total number of points.

8.3 Single Puncture Scaling with FMR

Figure 8.3 shows the strong scaling of simulations of a single puncture at rest with Fixed Mesh Refinement (FMR). All simulations were carried out to

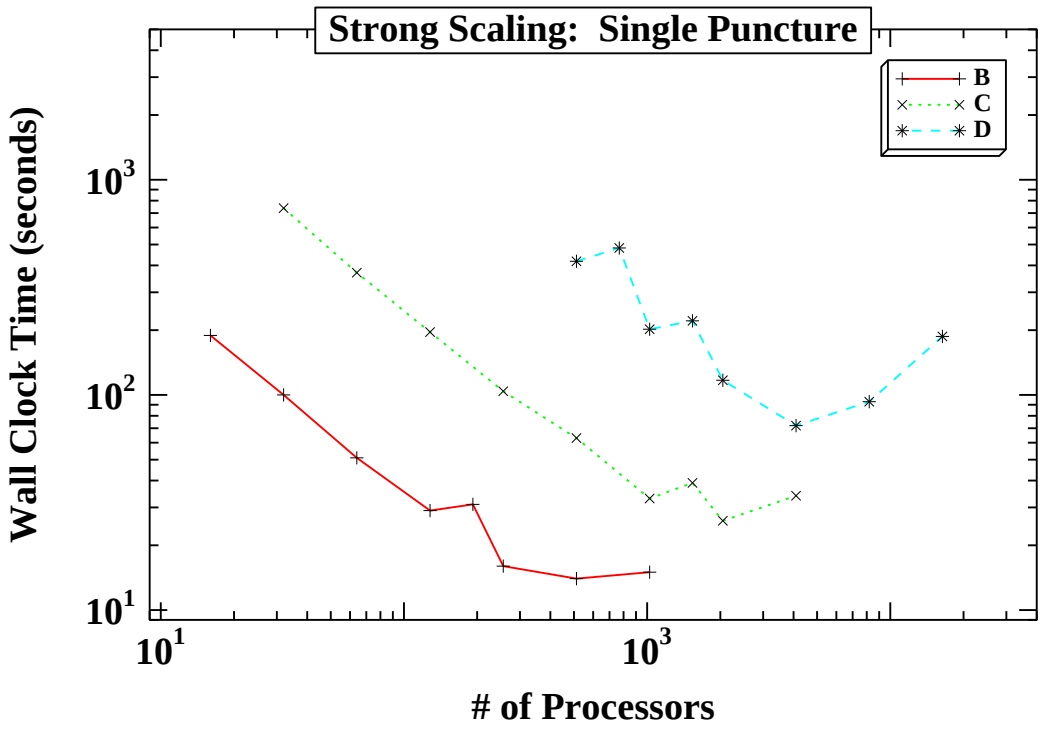


Figure 8.1: Strong scaling for runs of jobs B, C, and D outlined in Table 8.1. Each curve represents the same job being carried out on varying numbers of processors.

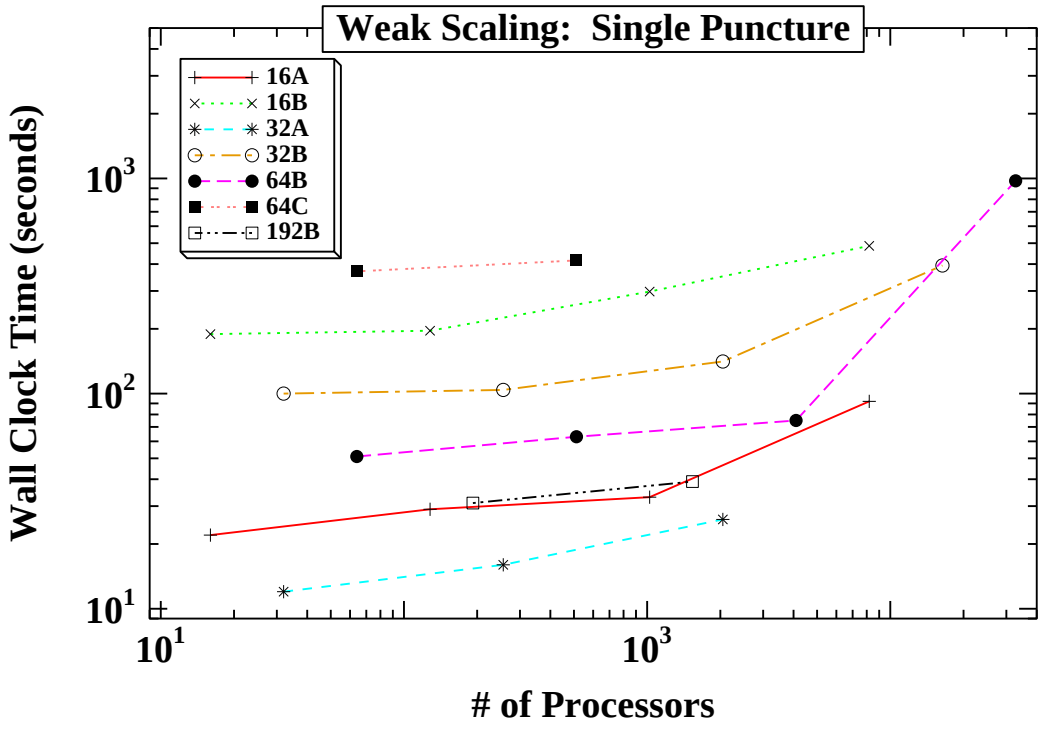


Figure 8.2: Weak scaling simulations where each subsequent data point on curve represents a job containing eight times the number of points carried out on eight times the number of processors (see Table 8.1). The labels for the curves given in the key define the number of processors and the job being run for the leftmost data point on the curve. For instance, on the curve 64B, the leftmost data point is 64 processors running job B. The next point on the curve is 512 processors running job C. The next two points on the curve are 4,096 processors running job D and 32,768 processors running job E, respectfully.

$t = .2M$, where M is the mass of the puncture. Details of the simulations are given in Table 8.2. Ideal scaling is a slop of -1. Reasonably close to ideal scaling is obtained up to $\sim$ 1000 processors. Over the range of 16 to 192 processors, the slope of curve A is $-.73$. Over the range of 256 to 1024 processors, the slope of curve B is $-.82$. However, the slope between 4096 and 8192 processors on curve C is only $-.35$.

Figure 8.4 shows weak scaling for the single puncture simulations with nine levels of fixed mesh refinement (see Table 8.2). The average slope of all three curves combined is .57. The result is consistent with the just described strong scaling results. We conclude comparing Figures 8.1 and 8.2 to Figures 8.3. and 8.4 that features of the mesh refinement introduce non-scaling behavior into the simulations. We are working with TACC (Texas Advanced Computing Center) and the SAMRAI tem to address this deficiency.

8.4 Scaling of Two Punctures with FMR

Figure 8.5 shows the strong scaling of simulations of two punctures, initially at rest, which then infall. All simulations were carried out to $t = .2M$. Here, M is the sum of the masses of both punctures. Details of the simulations are given in Table 8.3. Ideal behavior is a slope $b = -1$. Over the range of 16 to 192 processors, the slope of curve A is -.63. Over the range of 192 to 1536 processors, the slope of curve B is -.50. For curve C, which ranged from 4,096 to 16,384 processors, the average slope was -.03. These results show consistent behavior up to $\sim 1.5k$ processors, reflecting the lack of ideal but acceptable

Evolved to $t = .2M$			
Physical Domain	$[-100M, 100M]$ in x,y,z		
# Levels of Refinement	9		
Job	A	B	C
Resolution of Coarsest Grid	$4M$	$2M$	M
Resolution of Finest Grid	$\frac{M}{64}$	$\frac{M}{128}$	$\frac{M}{256}$
Strong Scaling:			
Processor Range	$16 - 192$	$256 - 1024$	$4096 - 8192$
Slope	-.73	-.82	-.35
Weak Scaling:			
Average Slope	.57		

Table 8.2: Simulations of a single puncture at rest with nine levels of mesh refinement. Note that the three jobs (A,B,C) are not quite factors of eight larger than each other in regards to the total number of points. Each subsequent job does, however, have twice the resolution everywhere throughout the domain. Each level is refined by a factor of two. M is the mass of the puncture.

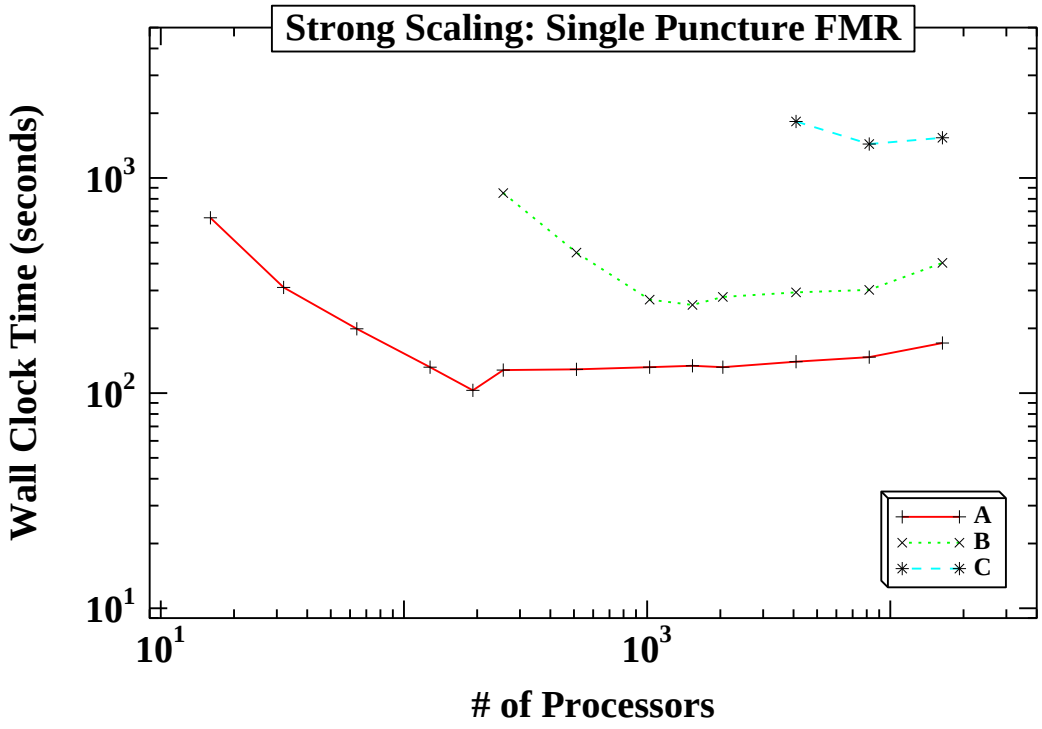


Figure 8.3: Strong scaling for runs of jobs A, B and C outlined in Table 8.2. Each curve represents the same job being carried out on varying numbers of processors.

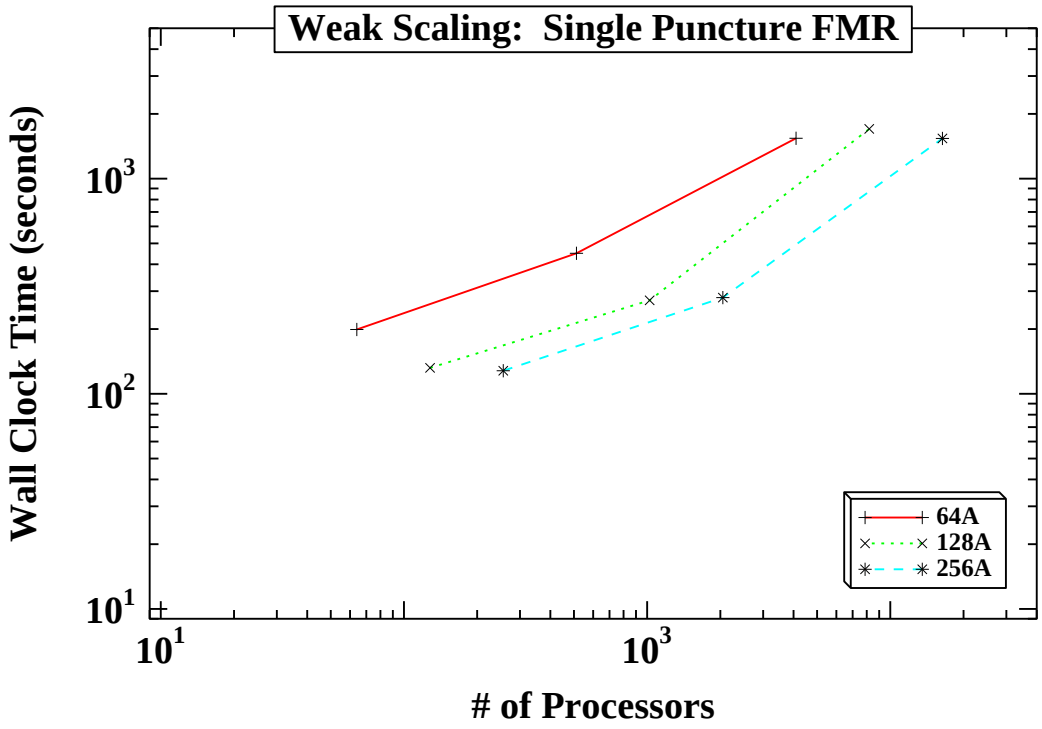


Figure 8.4: Weak scaling simulations where each subsequent data point on curve represents a job containing roughly eight times the number of points carried out on eight times the number of processors (see Table 8.2). The labels for the curves given in the key define the number of processors and the job being run for the leftmost data point on the curve. For instance, on the curve 256A, the leftmost data point is 256 processors running job A. The next point on the curve is 2048 processors running job B, followed by 16,384 processors running job C.

scaling already seen in the single puncture simulations. For those jobs, runs greater than $4k$ processors clearly have no scaling advantage.

Figure 8.6 shows weak scaling for the two puncture simulations (see Table 8.3). Ideal scaling behavior is zero slope. The average slope of all three curves in Figure 8.6 combined is .62. The slip is consistent with the single puncture FMR results of Figures 8.3 and 8.4.

Head-on collision Evolved to $t = .2M$			
Physical Domain	$[-100M, 100M]$ in x,y,z		
Puncture Locations	$(-\frac{3M}{2}$,0,0) and $(\frac{3M}{2}$,0,0)		
Initial Velocities	$v_1 = 0$ and $v_2 = 0$		
# Levels of Refinement	9		
# Levels Tracking Holes	2		
Job	A	B	C
Resolution of Coarsest Grid	$4M$	$2M$	M
Resolution of Finest Grid	$\frac{M}{64}$	$\frac{M}{128}$	$\frac{M}{256}$
Strong Scaling:			
Processor Range	$16 - 192$	$192 - 1536$	$4096 - 16384$
Slope	-.63	-.5	-.03
Weak Scaling:			
Average Slope	.62		

Table 8.3: Simulations of two equal-mass punctures initally at rest with nine levels of mesh refinement, the two finest of which are moving boxes that track the punctures. M is the total mass of the two punctures. Note that the three jobs (A,B,C) are not quite factors of eight larger than each other in regards to the total number of points. Each subsequent job does, however, have twice the resolution everywhere throughout the domain. Each level is refined by a factor of two.

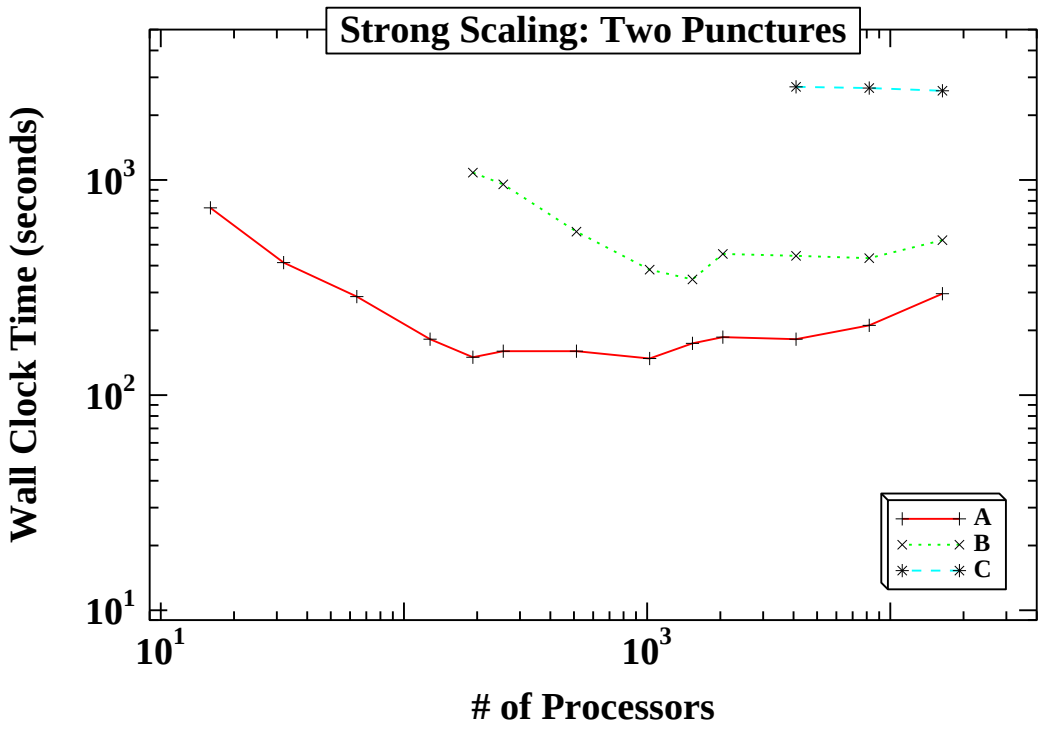


Figure 8.5: Strong scaling for runs A, B and C outlined in Table 8.3. Points on a given curve represent the same job for a varying number of processors.

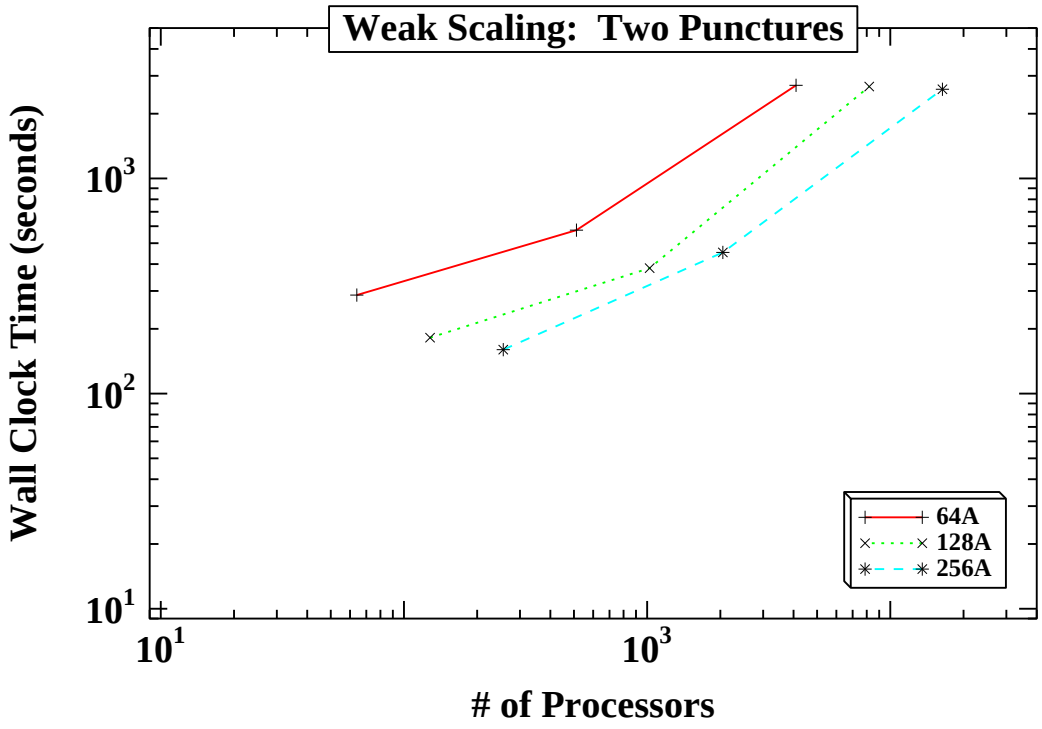


Figure 8.6: Weak scaling simulations where each subsequent data point on curve represents a job containing roughly eight times the number of points carried out on eight times the number of processors (see Table 8.3). The labels for the curves given in the key define the number of processors and the job being run for the leftmost data point on the curve. For instance, on the curve 256A, the leftmost data point is 256 processors running job A. The next point on the curve is 2048 processors running job B, followed by 16,384 processors running job C.

8.5 Equal Mass, Head-on Collision

Head-on collision simulation	
Physical Domain	$[-100M, 100M]$ in x,y,z
Puncture Locations	$(-\frac{3M}{2}$,0,0) and $(\frac{3M}{2}$,0,0)
Initial Velocities	$v_1 = 0$ and $v_2 = 0$
# Levels of Refinement	9
Resolution of Coarsest Grid	$4M$
Resolution of Finest Grid	$\frac{M}{64}$

Table 8.4: Simulation of two punctures initally at rest with nine levels of mesh refinement, the two finest of which are moving boxes that track the punctures. Each level is refined by a factor of two.

Table 8.4 provides the details of a simulation of a head-on collision carried out on 64 processors and allowed it to run for 24 hours. This is the same as Job A in Table 8.3. There are nine levels of mesh refinement, the finest two levels being moving boxes that track the punctures. Figure 8.7 shows the lapse on the initial slice ($t = 0M$). The two punctures merged from $t = 15M$ to $t = 17M$. This simulation ran to $t = 38M$ in 24 hours. This is a full scale demonstration of the *openGR* system, showing we can evolve from initial data through merger. Wave extraction is the next logical step. It has been implemented in *openGR* but has not been completely tested.

For a similar simulation, snapshots of before and after the merger are shown in Figure 8.8. When the two finest grids come into contact, they merge to form one grid. This is internally handled by *SAMRAI*.

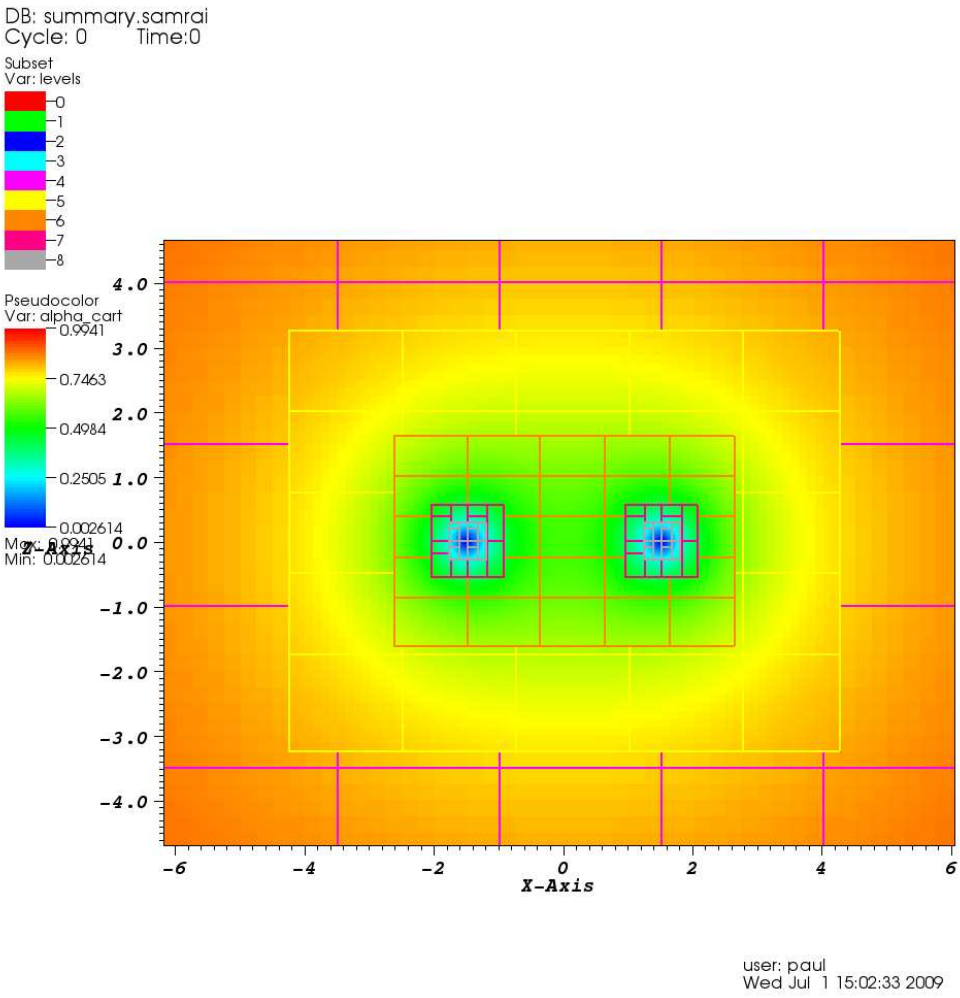
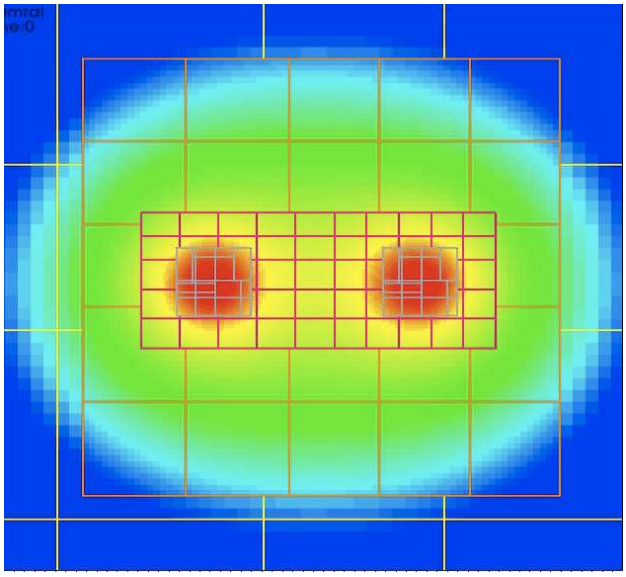
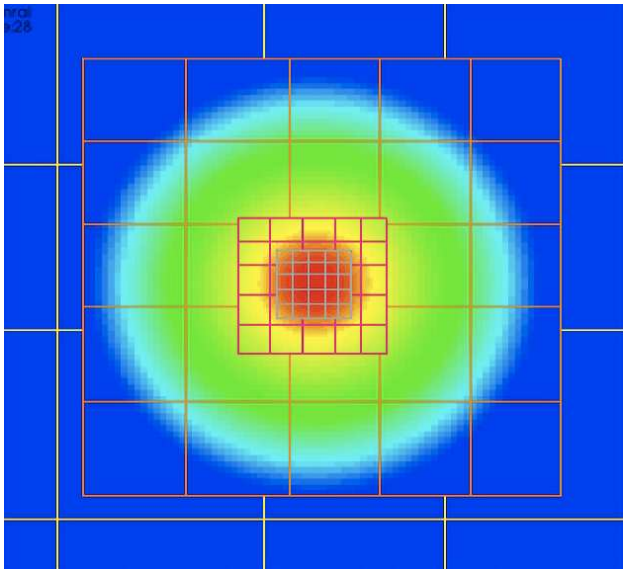


Figure 8.7: Zoomed in view of the lapse at the start of the simulation of two equal mass punctures initially at rest. The simulation outlined in Table 8.4 run on 64 processors for 24 hours, reaching time $t = 38M$. Merger took place from $t = 15M$ to $t = 17M$. The processor layout of the various refinement levels is also shown with the two finest levels tracking the holes. The units on the axes are in terms of the mass of a single puncture (half the total mass $M = M_1 + M_2$). Thus, the punctures are located at $(-\frac{3M}{2},0,0)$ and $(\frac{3M}{2},0,0)$ and their separation is $3M$. This is a plot of the lapse function α, which determines the redshift.



(a) Prior to merger



(b) After merger

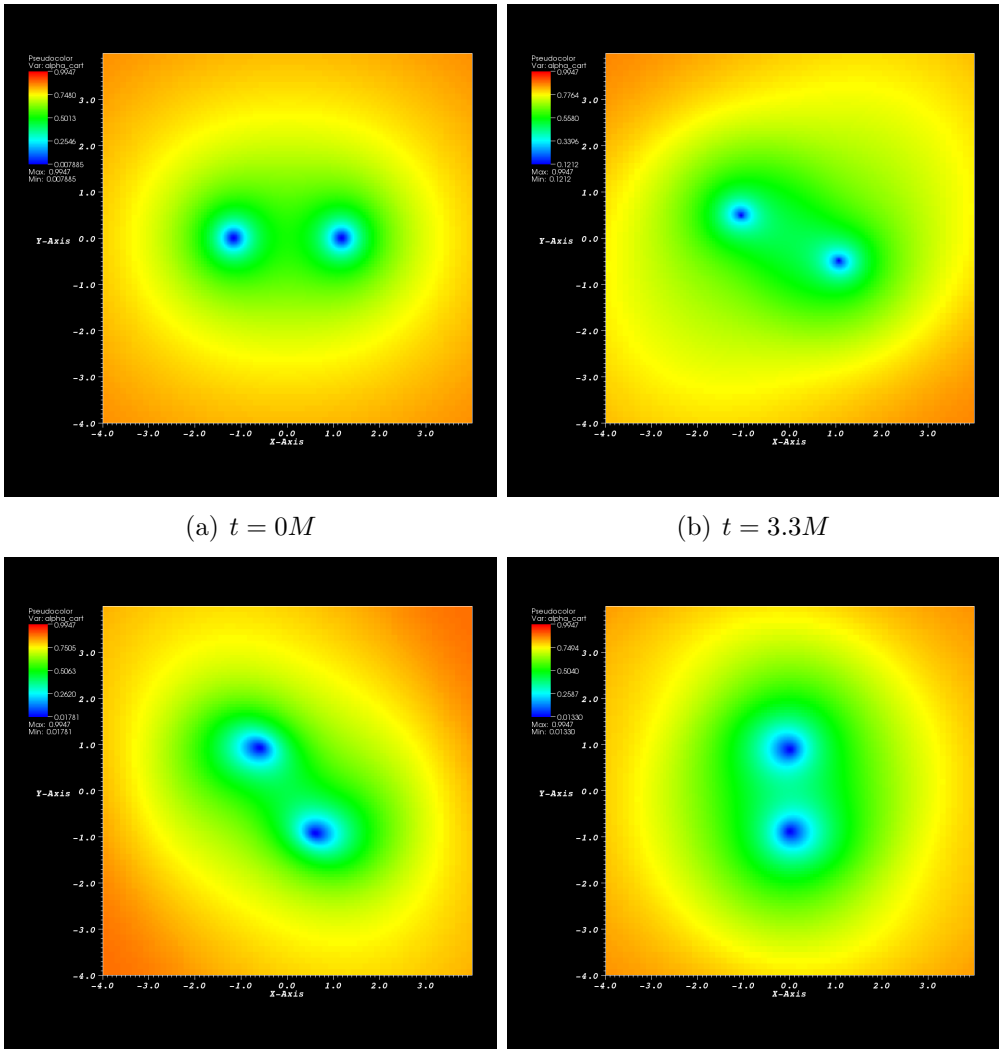
Figure 8.8: *openGR* simulation before and after merger with a maximum resolution of $\frac{M}{64}$. Surrounded by two patches that move with each puncture, and 7 more levels of fixed mesh refinement (all levels coarsened by a factor of two). The unusual layout of the finest grids in (a) is due to *SAMRAI's* load balancing of the 64 processors used for the simulation. The function plotted is the conformal factor ϕ (Eq. 5.5).

8.6 Initial Attempt at QC0

QC0 (quasi-ciricular) is the standard test run for binary black hole mergers [23]. The parameters of the simulation are found in Table 8.5. The simulation has run to $t \sim 14M$ in a time of 14 hours. Some of that time however was spent in the initial data solver. A typical run of QC0 evolves for approximately one orbit. The initial momentum we gave to the punctures was a bit too high, and this simulation will likely evolve to just beyond on orbit. This simulation does demonstrate that *openGR* is capable of evolving general binary black hole spacetimes. Figure 8.9 show the value of the lapse at various times in the evolution (between $t = 0M$ and $t = 10M$).

QC0	
Number of FMR levels	7
Coarsest level	$2M$
Finest level	$\frac{M}{32}$
Number of Processors	512
$\pm\frac{X}{M}$	1.169
$\frac{m}{M}$	.453
Lifetime of simulation	$14M$ and still evolving

Table 8.5: A first attempt at the simulation QC0, a standard test for evolving binary black holes. The simulation is being carried out on 512 processors. $\pm\frac{X}{M}$ is the inital coordinate positions of the two punctures. The initial momentum we entered appears to actually be a bit too high, and so the simulation will likely evolve for more than one orbit before merger. However, this simulation demonstrates that *openGR* has the ability to simulate general binary black hole mergers.



(a) $t = 0M$ (b) $t = 3.3M$

(c) $t = 6.7M$ (d) $t = 10.0M$

Figure 8.9: A first attempt at simulating QC0. The lapse function is show for different times in the evolution. See Table 8.5 for details.

8.7 Memory

Memory is the main bottleneck for the amount of resolution (and thereby accuracy) for simulations with a large enough domain to extract gravitational waves. All simulations reported in this work were carried out on *Ranger* at the Texas Advanced Computing Center (TACC). *Ranger* has 2 GB of memory per processing core.

We can investigate *openGR*'s memory performance by revisiting the scaling runs in Sec. 8.2 - 8.4. For the various simulations we can calculate the number of points (or cells) that are being evolved. More points are certainly contributing to the overall memory usage, such as the outer boundaries (5 points per side of the computational domain) and the ghostzones used for communication between SAMRAI patches. Load balancing with SAMRAI consists of breaking the computational domain into smaller patches, where each patch has 5 ghostzone points on a side. If the side of a patch is on the edge of the computational domain, it will have 5 outer boundary points instead of ghostzones. In general, SAMRAI load balancing consists of creating on the order of 8 patches per processor for a typical simulation that uses multiple levels of FMR. In the results that follow, FMR simulations will have considerably fewer points evolved per processor than unigrid runs. Improving the load balancing of SAMRAI will be of key importance moving forward to improve the scalability of *openGR*. Currently, FMR simulations are using too much of the memory for communication overhead in the form of ghostzones. One is not able to compute the number of ghostzones without knowing the

number of patches used and their shape. However, the ratio of ghostzones to evolved points does go up significantly when using multiple levels of FMR.

Table 8.7 displays the computational grid points available per processing core for the single puncture unigrid runs in Sec. 8.2. The evolved points available per processor are computed by investing how many processors it takes to evolve a given job. For instance, 32 processors can evolve Job C, but 16 processors cannot due to lack of memory (see curve C of Figure 8.1). As a result, the numbers reported give a lower limit (but close to the upper limit) for the number of points that can be evolved for a given number of processors. 16 and 64 were treated differently where the limits for a maximum number of evolved points is found by conducting runs with an increasing number of points until it crashes. The numbers reported (120^3 for 16 processors and 180^3 for 64 processors) are the total number of points for the last evolution before reaching the breaking point were the runs crash due to lack of memory. Regarding the overall picture, the number of evolved points per processor makes a steady decline as the number of processors increase.

Table 8.7 sums up the memory results for the scaling simulations carried out in Sec. 8.3 (see curves B an C in Figure 8.3). Those simulations consisted of evolving a single non-moving puncture spacetime using 9 levels of fixed mesh refinement. The number of points listed consist of one level of 100^3 points and another 8 levels of 51^3 points each. The number of evolved points per processor is low to begin with for these FMR simulations, and as expected gets lower still when increasing the size of the job (and number of processors).

Single Puncture Unigrid Simulations		
	Total size (grid points)	
Job A	40^3	
Job B	80^3	
Job C	160^3	
Job D	320^3	
Job E	640^3	
# processors	Job (or maximum points)	$\frac{points}{processor}$
16	120^3	47.6^3
32	C	50.4^3
64	180^3	45^3
512	D	40^3
32,768	E	20^3

Table 8.6: The number of points per processor for the simulations outlined in Table 8.1. The points per processor shown above provide a lower limit (but close to the actual limit) for the size of a job that can be run on the given amount of processors for a unigrid simulation. The points per processor are not counting ghostzones or outer boundary points.

Table 8.8 displays the results for the number of evolved points per processor available for the scaling simulations outlined in Sec. 8.4 (see curves B and C in Figure 8.5). Those simulations consisted of a spacetime with two punctures initially at rest that will then infall towards each other. The simulations were evolved for such a short time, $t = .2M$ (where M is the sum of the puncture masses), that the punctures hardly moved. Nonetheless, in those simulations the two finest grids (each puncture with its own set) are allowed to track each puncture. While we found in Sec. 8.4 that the moving meshes hurt the scaling performance relative to the case of 9 levels of FMR without moving grids, the moving meshes do not have any apparent memory overhead because we find very similar values for the points per processor in Tables 8.7 and 8.8. As an example of the grid layout, the number of points for Job B consists of 100^3 points on the coarsest level, 4 levels of 51^3 points, a level of 67×51^2 points, a level of 83×51^2 points, and two levels of $35^3 \times 2$ points.

Single Puncture 9 FMR Levels - No Moving Meshes		
	Total size (grid points)	
Job A	$250,000$	
Job B	$2,061,208$	
Job C	$16,741,816$	
# processors	Job	$\frac{points}{processor}$
256	B	20^3
4096	C	16^3

Table 8.7: The number of points per processor for the simulations outlined in Table 8.2. The points per processor shown above provide a lower limit (but close to the actual limit) for the size of a job that can be run on the given amount of processors for a simulation using 9 levels of FMR without any moving meshes. The points per processor are not counting ghostzones or outer boundary points.

Two Punctures 9 FMR Levels - Two Levels Move		
	Total size (grid points)	
Job A	$253,402$	
Job B	$2,092,254$	
Job C	$17,006,470$	
# processors	Job	$\frac{points}{processor}$
192	B	22.2^3
4096	C	16.1^3

Table 8.8: The number of points per processor for the simulations outlined in Table 8.3. The points per processor shown above provide a lower limit (but close to the actual limit) for the size of a job that can be run on the given amount of processors for a simulation with 9 levels of FMR where the two finest levels are allowed to move. The points per processor are not counting ghostzones or outer boundary points.

Chapter 9

Conclusion

openGR is the result of efforts by a number of researchers over several years. As I have reported here, openGR now contains the implementation of a full set of capabilities to be a useful production tool in studying the gravitational radiation production in black hole encounters, the strongest likely source for current and future gravitational radiation detectors.

Some of openGR's features, particularly wave extraction, are implemented but not thoroughly tested to assure their accuracy. That however should be a quick process now that the central code is functional. The demonstration of this functionality is in Figure 8.9 which show data similar to the test case QC0 which performs about an orbit before merging into a single ringing-down black hole [23].

The NSF has recently committed a substantial amount of conmputer time, on a very short timescale, (completion July 1 2010), for numerical relativists to develop new simulations to be used as templates for LIGO detection

searches. I am confident that openGR will be fully enabled in time to contribute to this effort.

Bibliography

[1] Matt Anderson, "Constrained Evolution in Numerical Relativity", Ph.D. Thesis, The University of Texas at Austin, (2004).

[2] E. Newman, R. Penrose, *J. Math. Phys.* **3**, 566, (1962).

[3] Andrea Nerozzi, "Towards BH-NS Merger Simulations: Initial Data, Evolution and Wave Extraction" Ph.D. Thesis,

[4] B. Kelly, The Maya Project: The Newman-Penrose Scalars - http://www.eng.uah.edu/ jacksoa/literature/NPScalars.pdf.

[5] M. Alcubierre, *Introduction to 3+1 Numerical Relativity* (Oxford Science Publications, 2008).

[6] B. Schutz, *A First Course in General Relativity, 2nd ed.*, (Cambridge University Press, 2009).

[7] A. Bechinger, "Gravitational Wave Extraction in Numerical Relativity", Master's Thesis, University of Texas at Austin, 2005.

[8] A. Nerozzi, O. Elbract, "Using curvature invariants for wave extraction in numerical relativity", (2008). e-Print: arXiv:0811.1600 [gr-qc].

[9] A. Nerozzi, "Scalar functions for wave extraction in numerical relativity", *Phys. Rev. D.* **75**, 104002, (2007).

[10] Ambramovici et al., "LIGO: The Laser interferometer gravitational wave observatory", Science **256**, 325 (1992).

[11] B. Caron et al., "The Virgo Interferometer", Class. Quant. Grav. **14**, 1461 (1997).

[12] H. Lück et al., "The Geo-600 Project", Class. Quant. Grav. **14**, 1471 (1997).

[13] M. Ando et al., "Stable Operation of a 300-m Laser Interferometer with Sufficient Sensitivity to Detect Gravitational Wave Events Within our Galaxy", Phys. Rev. Lett. **86**, 3950 (2001).

[14] F. Pretorius, "Evolution of Binary Black Hole Spacetimes", Phys. Rev. Lett. **95**, 121101 (2005).

[15] M. Campanelli, C.O. Lousto, P. Marronetti and Y. Zlochower, "Accurate Evolutions of Orbiting Black-Hole Binaries Without Excision", Phys. Rev. Lett. **96**, 111101 (2006).

[16] J.G. Baker, J. Centrella, D. Choi, M. Koppitz and J. van Meter, "Gravitational Wave Extraction from an Inspiraling Configuration of Merging Black Holes", Phys. Rev. Lett. **96** 111102 (2006).

[17] B. Brügmann, J.A. González, M. Hannam, S. Husa and U. Sperhake, "Calibration of Moving Puncture Simulations", Phys. Rev. D. **77**, 024027, (2008).

[18] Wissink, A. M., R. D. Hornung, S. R.Kohn. S. S. Smith, and N. S. Elliott, "Large Scale Structured AMR Calculations Using the SAMRAI Framework", SC01 Proceedings, Denver, CO, Nov. 10-16, 2001. Also available as LLNL technical report UCRL-JC-144755.

[19] M. Ansorg, "A double-domain spectral method for black hole excision data", Phys. Rev. D **72**, 024018 (2005).

[20] T. Regge and J. Wheeler. "Stability of a Schwarszchild singularity." Phys. Rev., 108(4):10631069, 1957.

[21] F. J. Zerilli. "Gravitational field of a particle falling in a Schwarzschild geometry analyzed in tensor harmonics." Phys. Rev. D., 2:2141, 1970.

[22] S. A. Teukolsky. "Perturbations of a rotating black hole. I. fundamental equations for gravitational, electromagnetic, and neutrino-field perturbations." Astrophys. J., 185:635647, 1973.

[23] J. Baker, M. Campenelli, C.O. Lousto, R. Takahashi, "Modeling gravitational radiation from coalescing binary black holes", Phys. Rev. D **65**, 124012, (2002).

[24] J. W. York and T. Piran. *The initial value problem and beyond.* In Richard

A. Matzner and L.C. Shepley, editors, *Spacetime and geometry: the Alfred Schild lectures*, pages 147176. University of Texas Press, 1982.

[25] R. Kerr and A. Schild. Some algebraically degenerate solutions of Einsteins gravitational field equations. In *Applications of Nonlinear Partial Differential Equations in Mathematical Physics*, Symposium in Applied Mathematics Proceedings, v. 17., pages 199209. American Mathematical Society, 1965. Symposium held in New York City, (1964).

[26] Heinz Otto Kreiss and Omar E. Ortiz. Some mathematical and numerical questions connected with first and second order time dependent systems of partial differential equations. *Lecture Notes in Physics*, 604:359370, (2002).

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

[28] M. Shibata and T. Nakamura, Phys. Rev. D **52**, 5428 (1995).

[29] T.W. Baumgarte and S.L. Shapiro, Phys. Rev. D **59**, 024007 (1999).

[30] D.S. Brill and R.W. Lindquist, Phys. Rev. **131**, 471 (1963).

[31] J. M. Bowen and J. W. York, Phys. Rev. D **21**, 2047 (1980).

[32] C. Bona, J. Massó, E. Seidel, and J. Stela, Phys. Rev. D **56**, 3405 (1997)

[33] J.G. Baker. J. Centrella, D.-I. Choi, M. Koppitz, and J. van Meter, Phys. Rev. D **73**, 104002 (2006).

[34] J.R. van Meter, J.G. Baker, M. Koppitz, and D.-I. Choi, Phys. Rev. D **73**, 124011 (2006).

[35] C. Gundlach and J.M. Martin-Garcia, Phys. Rev. D **74**, 024016 (2006).

[36] M. Alcubierre, G. Allen, G. Lanfermann, D. Rideout, "Boundary Conditions," (2003).

[37] A. Sommerfeld, *Partial Differential Equations in Physics*, Academic Press, New York, New York, 1949.

Vita

Paul Walter graduated from Centennial High School in Pueblo, Colorado, in 1997. He then attended the University of Notre Dame in Notre Dame, Indiana, majoring in physics and graduating with a Bachelor of Science degree in 2001. In August of 2002, Paul then entered graduate school at the University of Texas at Austin Department of Physics.

Permanent Address: 2501 Manor Rd. Apt 306

Austin, TX 78722

This dissertation was typeset with LaTeX 2_ε[1] by the author.

[1] LaTeX 2_ε is an extension of LaTeX. LaTeX is a collection of macros for TeX. TeX is a trademark of the American Mathematical Society. The macros used in formatting this dissertation were written by Dinesh Das, Department of Computer Sciences, The University of Texas at Austin, and extended by Bert Kay, James A. Bednar, and Ayman El-Khashab.