ABSTRACT

Title of Dissertation:	Numerical studies of constraints and gravitational wave extraction in general relativity
	David Robert Fiske, Doctor of Philosophy, 2004
Dissertation directed by:	Professor Charles W. Misner Department of Physics

Within classical physics, general relativity is the theory of gravity. Its equations are non-linear partial differential equations for which relatively few closed form solutions are known. Because of the growing observational need for solutions representing gravitational waves from astrophysically plausible sources, a subfield of general relativity, numerical relativity, has a emerged with the goal of generating numerical solutions to the Einstein equations. This dissertation focuses on two fundamental problems in modern numerical relativity: (1) Creating a theoretical treatment of the constraints in the presence of constraint-violating numerical errors, and (2) Designing and implementing an algorithm to compute the spherical harmonic decomposition of radiation quantities for comparison with observation.

On the issue of the constraints, I present a novel and generic procedure for incorporating the constraints into the equations of motion of the theory in a way designed to make the constraint hypersurface an attractor of the evolution. In principle, the prescription generates non-linear corrections for the Einstein equations. The dissertation presents numerical evidence that the correction terms do work in the case of two formulations of the Maxwell equations and two formulations of the linearized Einstein equations.

On the issue of radiation extraction, I provide the first in-depth analysis of a novel algorithm, due originally to Misner, for computing spherical harmonic components on a cubic grid. I compute explicitly how the truncation error in the algorithm depends on its various parameters, and I also provide a detailed analysis showing how to implement the method on grids in which explicit symmetries are enforced via boundary conditions. Finally, I verify these error estimates and symmetry arguments with a numerical study using a solution of the linearized Einstein equations known as a Teukolsky wave. The algorithm performs well and the estimates prove true both in simulations run on a uniform grid and in simulations that make use of fixed mesh refinement techniques.

Numerical studies of constraints and gravitational wave extraction in general relativity

by

David Robert Fiske

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2004

DEDICATION

To my parents, who always wanted the best for me.

Y a Vili, quien justamente es así.

ACKNOWLEDGEMENTS

My time as a graduate student has been split into three "epochs," and I wish to acknowledge the generosity of people with whom I have interacted in each.

Most recently, I have spent over a year working with the numerical relativity group at Goddard Space Flight Center. I am most grateful to Joan Centrella, who has been kind enough to take me into her group at a late stage in my graduate career and to helped me to define much of the work that constitutes Chapter 4 of this dissertation. Her efforts have helped to ensure my timely graduation. In the actual execution of this plan, John Baker has provided key guidance both by providing references to the literature and by serving as a sounding board for many of my ideas. I am grateful to James van Meter for continually catching my bugs and keeping a good sense of humor while doing so, and to Breno Imbiriba and Dae-Il Choi for showing me the ropes at Goddard. David Brown has been a source of expertise on several topics, and also made Figure A.1. Richard Matzner has contributed valuable comments on early drafts of the dissertation and has been a valuable resource as I have attempted to round out the material in the text. Dan Brennan, Kim Engle, Phil Newman, Josephine Palencia, and Jeff Simpson provided essential services administering the local Beowulf clusters.

Prior to my time at Goddard, I spent most of a year working primarily alone at the University of Maryland. During this time I benefited greatly from discussions with David Mattingly and Ted Jacobson, primarily on the work that formed the basis for Chapter 3 of this dissertation. I am also grateful to Ted for showing a continuing interest in my success and well-being during my time as a student at Maryland. Manuel Tiglio at the Louisiana State University was also kind enough to take an interest in my work and has provided valuable comments and insights on many points.

I spent most of 2002 at the Center for Gravitational Physics and Geometry at the Pennsylvania State University. Pablo Laguna and Deirdre Shoemaker were my gracious hosts, and did their best to ensure that I felt part of the Center and benefited from what it has to offer. Ken Smith and Bernard Kelly were my companions in this part of the journey; in addition to their great work, they made the time away from home more pleasant. Erik Schnetter, although no longer at Penn State by the time I arrived, taught me a lot about good code design by phone, by email, and most of all by the example he left behind in his Maya code. He was also kind enough to play host during two of my trips to Germany.

Some people, of course, do not belong to any of these periods in particular. Charles Misner has, from the beginning, provided valuable advice regarding my work, and perhaps more importantly has provided me with many opportunities to interact with the wider community. This includes coordinating my time at Penn State and Goddard, but also includes sending me to conferences both in the U.S. and abroad. I also grateful to Conrad Schiff for help at several stages during my time at Maryland, in addition to the occasional free meal. Alex Dragt advised me during my first year of graduate school, and I am grateful for his support and direction.

Outside of school I have received a lot of support from family and friends. My parents have always done what they could to support me morally and emotionally. Violeta Prieto has been a great friend throughout my time at Maryland, and has tolerated the bad moods and odd work schedules that have been part of finishing this dissertation with love and support (not to mention meals, laundry, and a detailed proofing of the text). Matt Ferguson, Liz Hays, Chen Ling, and especially Sarah Donnelly and Rachel Grubbs have been close companions throughout school.

Finally, but not least, I have been fortunate to have great teachers and professors from elementary school through graduate school. Certainly I cannot name them all, but I wish to acknowledge especially Alvin Bell, Vitaly Bergelson, Ron Bowerman, Barbara Buttermore, Mark Dickman, Richard Furnstahl, Mike Gilligan, John Givens, Roger Gossman, Cora Kerr, Albert Laux, Lisa Marker, Dan Matheny, Beth Niemeyer, LeEddy Smith, Lisa Snook, Linda Straley, and Judy Withrow. In addition to being excellent teachers, these individuals enriched my experience either by sharing their free time to provide additional opportunities, by encouraging and allowing me to go beyond classroom material (often by allowing me to choose challenging homework problems myself rather than compelling me to do those assigned to the class), or, in many cases, both.

This work has been supported by a University of Maryland Graduate Fellowship, U.S. Department of Energy grant DEFG02-96ER40949, U.S. National Science Foundation grant PHY-0071020, and NASA Space Sciences grant ATP02-0043-0056. I was fortunate to twice received travel support from Sonderforschungsbereich 382 funded by the DFG and headed by Hanns Ruder at Theoretische Astrophysik Tübingen in Germany. Some of my computations were performed on Beowulf clusters at NASA Goddard Space Flight Center operated by the Space Science Data Operations Office and the Commodity Cluster Computing Group.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

This dissertation is concerned with constructing numerical solutions to Einstein's theory of general relativity and with developing tools for analyzing those solutions in a way compatible with observational attempts to detect gravitational radiation. This endeavor draws from and contributes to the areas of mathematical physics, numerical analysis, computer science, and astrophysics. Indeed, while the tone of the dissertation, both in terms of the test problems presented and the background material covered, centers around general relativity, most of the original work in the dissertation has applications outside of numerical relativity. The constraint driver terms detailed in Chapter 3, for example, are applicable to the numerical evolution of any system of partial differential equations which separate into constraint and evolution equations. In more physical language, this means that the algorithm constructed in that chapter is applicable to any gauge theory, of which there are many in modern physics. Likewise, the analysis of Misner's novel algorithm for computing spherical harmonic components of functions represented on a cubic grid, which is central to Chapter 4, has applications extending even beyond traditional physics.

Of course the emphasis of this dissertation is rooted in numerical relativity. While the prescription for generating constraint driver terms, alluded to above, is not specific to general relativity, I study this new computational tool on examples from linearized gravity and on formulations of the Maxwell equations known to have properties analogous to the Einstein equations. After constructing generally applicable error estimates for Misner's algorithm, I demonstrate that the algorithm works and that the error estimates are true with numerical simulations of a pure gravitational wave spacetime. The simulation, moreover, runs inside a code designed to solve the full, non-linear Einstein equations with quite generic initial data and which uses fixed mesh refinement technology to efficiently allocate computer resources. This is a code that implements many the most important technological tools for solving the Einstein equations in situations of interest to the experimental community.

The tests here are not only focused on problems for numerical relativity, they are implemented such that they form a natural part of a consistent and continuing effort to invent, implement, and improve techniques for computing waveforms and other interesting properties of strongly gravitating bodies.

The dissertation is organized as follows: In the remainder of this chapter, I provide some historical context and motivation for the work that I have done, and I also briefly describe the notation and conventions employed in the main text. In Chapter 2, I provide a technical sketch of the mathematical foundations of separating the covariant Einstein equations, in which space and time are treated on equal footing, into spatial and temporal pieces. This material is introductory, but is much more technical than the introductory material in Chapter 1. In Chapter 2, I also introduce the notion of different formalisms of the equations. This leads directly into Chapter 3, where I study the role of the constraint equations of the theory in numerical evolutions. This chapter describes original work toward incorporating the constraint equations of the theory into the evolution equations in a way that is designed to force numerical evolutions to obey the constraints. I begin with an extended explanation of work that I previously published on the subject — an application of the procedure to the Maxwell equations — and augment that with more recent work in which I apply the procedure to gravitational plane waves. In Chapter 4, I switch directions and begin a study of gravitational radiation in numerical simulations. I sketch the mathematical structure, in this case the Newman-Penrose formalism, needed to sensibly define gravitational radiation. In practice, most people like to decompose wave signals into spherical harmonics components, so I then describe the appropriate notion of spherical harmonics, spin-weighted spherical harmonics, for radiation quantities; I describe in detail an algorithm previously published by Misner for computing spherical harmonics components of functions represented on a cubic grid; and I compute, for the first time, detailed estimates of the numerical errors incurred by the algorithm. This theory is then tested, with excellent results, in conjunction with a large-scale numerical relativity code. Results are presented. Finally I conclude the main text with some discussion in Chapter 5, and provide additional details that do not fit into the main body of the text in the Appendices.

1.1 Historical Context and Motivation

Einstein presented his theory of general relativity in 1915 [38]. Like his special theory of relativity, the new theory treated space and time as two parts of a single entity, spacetime. The theory asserts that gravity results from curvature in spacetime and that matter determines the spacetime curvature. This interplay between curvature and matter is described by the Einstein equation

$$G_{\mu\nu} = 8\pi T_{\mu\nu} \tag{1.1}$$

which is a non-linear, partial differential equation for the spacetime metric and the matter fields.[1] The symbol $G_{\mu\nu}$ represents the Einstein tensor and $T_{\mu\nu}$ is the stress-energy tensor for any matter in the spacetime. Both are symmetric and their indices run from 0 to 3. Because, in this dissertation, I work entirely on vacuum problems, I henceforth set $T_{\mu\nu} = 0$. Even with this simplification, however, there are few known analytic solutions to (1.1), and, of the known solutions, many are cosmological solutions which have no direct application to gravitational wave physics. To date, the two body problem, which is a problem of great interest in gravitational wave

[1]Throughout the dissertation I use geometric units $G = c = 1$, where G is Newton's gravitational constant and c is the speed of light. In these units length, time, and mass all have the same dimensions, which means that I will frequently measure time in units of length or length in units of mass!

physics, remains unsolved.[2]

There are some well known solutions to the full Einstein equations (1.1), including the Schwarzschild solution mentioned in Footnote 2. While these solutions played an extremely important role for people first sorting out the implications of the theory, they, for me in the context of this dissertation, are only convenient test cases. I will introduce the details of specific solutions in the text when needed. There is, of course, a rich history both to the construction and interpretation of these solutions; interested readers are referred to MTW [66] for more of this history. What is relevant to the current discussion is that the earliest solutions to the Einstein equations were constructed by considering the equations in their covariant form (1.1), often simplified by imposing certain convenient symmetries.

In contrast to this approach, numerical relativists would like to think of the Einstein equations as an initial value problem. Initial data should be specified and then evolution equations, in discretized form, should be used to advance that data to some later (or earlier) time. In order to follow such a program, however, one needs to break spacetime back into space and time. One such description of the Einstein equations, credited to Arnowitt, Deser, and Misner (ADM), was formulated in the late 1950s [9]. ADM found that the Einstein equations do not admit an initial value problem, but do admit a *constrained* initial value problem. Only six of the ten Einstein equations provide evolution information. The other four equations are constraints that the data must satisfy at all times.[3] The existence of the constraints

[2]One should keep in mind that the "two body problem" can actually be a problem in vacuum. The Schwarzschild solution, for example,

$$ds^2 = -\left(1 - \frac{2M}{r}\right) dt^2 + \left(1 - \frac{2M}{r}\right)^{-1} dr^2 + r^2 d\Omega^2$$

was discovered as early as 1916 [76], and is a solution to the vacuum Einstein equations. It nonetheless describes a black hole of mass M. It is the vacuum problem of computing the orbits of a binary black hole system, in fact, that is most frequently meant by the "two body problem," at least in the numerical relativity community.

[3]I will show in the text that it is sufficient to solve the constraint equations in the initial data provided that the *exact* (not discretized) evolution equations are integrated exactly.

is directly related to the fact that general relativity is a gauge theory — in the case of general relativity the gauge freedom corresponds to freedom in how spacetime coordinates are chosen. Chapter 2 deals with the decomposition of spacetime along the lines of ADM, and Chapter 3 deals with the constraints of the theory in great detail.

Arnowitt, Deser, and Misner were completing their work on this subject just before computers were marginally large enough, fast enough, and easy enough to use that one might consider trying to solve the ADM form of the Einstein equations numerically. The first documented attempt to do so appears to be work by Hahn and Lindquist in 1964 [47]. This work was quickly extended by Smarr [80, 81] and by Eppley [39]. These first attempts were centered around evolving Misner data [64] in axisymmetry (2+1 dimensions). Lack of sufficient computer resources would likely have impeded these efforts and those that followed them over the next twenty-plus years. A breakthrough finally came in the 1990s when, using faster computers with more memory, the Binary Black Hole Grand Challenge Alliance in the United States successfully simulated a head-on black hole collision. As a post-processing step they were also able to compute the event horizon in the spacetime. This work was still in axisymmetry [63].

Some of the first attempts to apply numerical relativity in three dimensions focused on single Schwarzschild black holes. This was an appealing testbed since the analytic solution was known. One of the earliest attempts to evolve a Schwarzschild black hole in three dimensions was published by Anninos el al. [8]. In their 1995 paper, the authors lament:

> Progress in three dimensional numerical relativity has been impeded in part by a lack of computers with sufficient memory and computational power to perform well resolved calculations of 3D spacetimes.

(As I will describe shortly, the situation, even with computer power increasing according to Moore's law, is not so much better now, nearly ten years later.) Important theoretical and technical advances were nonetheless made in the area of black hole simulations during the 1990s. Two different methods, both of which are still used today, were devised for handling the physical singularities inherent in black hole solutions — namely the *puncture* method [23, 24], which generalized the Brill-Lindquist prescription [19] for initial data of black holes at rest and which can be generalized to the Bowen-York prescription [18] for spinning and moving black hole initial data; and the *excision* method [3], in which a portion of the spacetime containing the singularity and interior to the event horizon is simply not evolved. Dynamical gauge conditions that would prevent hypersurfaces from reaching the singularity were also developed [17].

Black hole research continued into this decade with further advancements in gauge conditions [4, 1], the first evolutions of binary black hole systems that ran (stably) for a time comparable to an orbital period [25], and advances in excision techniques that allow black holes to move through the numerical grid [34, 79]. Even though progress has been made, state of the art simulations continue to push the limits of modern computer power.[4] It is not sufficient to wait for computers to become larger in order to increase resolution in numerical relativity simulations; the algorithms also must become more efficient. One example of this is the application of *mesh refinement* to numerical relativity computations. Mesh refinement means choosing the resolution of the simulation differently in different parts of the computational domain. It comes in two forms. In adaptive mesh refinement (AMR), the algorithm chooses how to distribute the available resources throughout the domain and redistributes the resources according to the dynamics of the evolution. In fixed mesh refinement (FMR) a person chooses how to distribute resources, and this distribution does not change during the course of the simulation. Both types of mesh refinement have long been practiced in hydrodynamics.

Choptuik, starting in the late 1980s, was the first to use AMR in numerical relativity in the course of his studies of critical collapse phenomena in scalar fields in one dimension [31, 32]; this work has been extended within the last year to two dimensions [33]. In two dimensions, AMR has also been used to study inhomogeneous

[4]Indeed it is likely that this statement will still be true in ten or twenty years because, as more computer resources become available, the problems attempted become larger!

cosmologies [50, 13], and was applied to studies of a Schwarzschild black hole [23]. Fixed mesh refinement has been used in short simulations of a binary black hole merger [24], a Schwarzschild black hole with excision [74], and orbiting, equal mass black holes in a co-rotating gauge [25]. The propagation of gravitational waves has been studied in AMR simulations of model equations describing perturbations of the Schwarzschild solution [70] as well as in the full, three-dimensional Einstein equations [68]. Detailed studies of how faithfully gravitational waves and other strong gradients pass through refinement boundaries have only begun quite recently. The first published results from numerical relativity appeared just this year, first in the case of pure waves [30] and then in the case of puncture black hole initial data [55]. Chapter 4 of this dissertation builds directly on the work of Refs. [30] and [55].

The first indirect observational evidence for gravitational waves came from observations of PSR 1913+16, the first discovered pulsar in a binary system [52, 53, 54]. Careful study of the orbital period of the binary, measured by observations of the time of arrival of the pulses showed that the system lost angular momentum and energy at a rate quite consistent with the hypothesis that it was radiating gravitational waves [35, 36]. Another pulsar-containing binary, PSR 1534+12, was discovered in 1991 by Wolszczan [92], and detailed measurements of it have also lead to stringent tests of strong-field general relativity [82, 83]. For a review of the early history of this topic, see the text of the Nobel lecture by Taylor [84] or by Hulse [51].

At the present time, several ground-based detectors, using laser interferometry, are already or very near to operating [11, 43, 75], and the NASA-ESA space-based antenna LISA was scheduled to be launched in 2011 [14]; budget cutbacks currently make a delay of one to two years likely [90]. These experiments should provide the first direct probe of strong-field gravitational physics. The data analysis needs of these observations, however, require accurate waveform templates for use in matched-filtering algorithms, and while the early and late stages of a merger process can be treated analytically using post-Newtonian and perturbation theory, respectively, the highly dynamical merger period can only be understood with the full, non-linear Einstein equations. In this regime, numerical relativity is essential [75]. In addition to the laser interferometers, several groups around the world continue to operate Weber bars, a technology pioneered by Joseph Weber at the University of Maryland [48]. While there is only a small chance of positively detecting a gravitational wave with such detectors, they have set observational upper bounds.

1.2 Notation and Conventions

This section describes notation in use throughout the text.

With respect to tensor indices, this dissertation follows the convention that Latin indices range over spatial values (1, 2, 3), while Greek indices range over spacetime indices (0, 1, 2, 3). Because most quantities in the dissertation are written in a 3+1 framework, most tensors that appear here will be spatial tensors. When the possibility of confusion arises, I prepend a "(4)" as a superscript; e.g. ${}^{(4)}R_{\mu\nu\alpha\beta}$ is the Riemann tensor associated with the spacetime metric, whereas R_{ijmn} is the Riemann tensor associated with the spatial metric. In a few cases, I also find it convenient to put spacetime indices on purely spatial tensors. In these cases all "time" components are understood to be identically zero. The Einstein summation convention, that repeated indices are summed over, is used except where I explicitly indicate that it is not.

Partial derivatives are interchangeably denoted

$$\frac{\partial f}{\partial x^i} = \partial_i f = f_{,i} \tag{1.2}$$

and covariant derivatives, likewise, are denoted

$$\nabla_i f = f_{;i} \tag{1.3}$$

according to convenience. The Lie derivative of a field f with respect to a vector v^a is denoted $\pounds_v f$. For variational derivatives, I adopt the notation[5]

$$\frac{\delta\phi(x')}{\delta\phi(x)} = \delta(x - x') \tag{1.4}$$

[5]This notation is consistent with Ref. [71] even though it does not appear, to my knowledge, in the general relativity literature.

where δ is a Dirac delta function. In most cases the operator will act on an integral, as in

$$\frac{1}{2}\frac{\delta}{\delta\phi(x)}\int\left[\phi(x')^2+\phi_{,x'}(x')^2\right]dx'=\phi(x)-\phi_{,xx}(x) \tag{1.5}$$

for example. In such cases, the operator is a mapping from functionals to functions. Note also that, as in the example, I freely integrate by parts without regard for the boundary integrals. This is equivalent, in practice, to assuming that the boundaries of my domains have appropriate boundary conditions. This assumption proves sound enough for the cases that I study here.

Finally, the metric signature will be $(-+++)$ for the spacetime metric, and, correspondingly, $(+++)$ for the spatial metric. I use the standard notation

$$t_{(ab)}=\frac{1}{2}\left(t_{ab}+t_{ba}\right) \tag{1.6}$$

and

$$t_{[ab]}=\frac{1}{2}\left(t_{ab}-t_{ba}\right) \tag{1.7}$$

and generalize it to more indices when needed. Signs and index ordering on conventional tensors (like Riemann, for example) are as in MTW [66].

Chapter 2

Separating Space from Time

The Einstein equations written in their usual form (1.1) are manifestly covariant. Space and time appear as equal partners. While this form pleases the theorist, it is not well suited for numerical simulations. When constructing a simulation, it is preferable to think of time separate from space, with the hope of specifying initial data for the spatial domain of a simulation and evolving that data forward in time according to some evolution equations.[1]

This chapter sketches the formal steps required to separate space from time in the Einstein equations. The results will be a *constrained* initial value formulation of the Einstein equations. That is, some of the Einstein equations will turn out to be constraints that the data must satisfy at all times, while others provide evolution equations for given initial data. After deriving the desired result from an appropriate Lagrangian in canonically conjugate variables, I introduce the notion of different "formalisms" of the equations, which plays a key role in modern attempts to build numerically stable simulations.

[1]This is not the only approach. While all numerical relativity codes of which I am aware do some sort of evolution, some authors have, for example, experimented with specifying initial data on surfaces that are, at least for part of the domain, light-like rather than space-like [91].

2.1 ADM Decomposition

The currently standard space plus time decomposition was popularized by Arnowitt, Deser, and Misner in the late 1950s [9] and has subsequently been referred to as either an "ADM decomposition" or a "3+1 split."[2] The discussion in this section is largely a review of the original ADM paper [9], supplemented by the discussion by Wald [89, Appendix E].

The starting point for the discussion is a spacetime $(M, {}^{(4)}g_{\mu\nu})$ comprised of a manifold M and a metric ${}^{(4)}g_{\mu\nu}$, and the Lagrangian

$$\mathcal{L} = {}^{(4)}R\sqrt{-{}^{(4)}g}. \tag{2.1}$$

This Lagrangian is associated the usual Einstein-Hilbert action from which the covariant form of the vacuum ($T_{\mu\nu} = 0$) Einstein equations (1.1) can be derived by varying the action with respect to the spacetime metric. In (2.1), the quantities on the right hand side are the determinant of the four dimensional spacetime metric and its Ricci scalar.

To proceed from this point, the degrees of freedom on spatial slices must be isolated from the degrees of freedom related to the passage of time. In their original work on the subject, ADM identified

$$\begin{aligned} g_{ij} &= {}^{(4)}g_{ij} && \text{(2.2a)}\\ \alpha &= (-{}^{(4)}g^{00})^{-1/2} && \text{(2.2b)}\\ \beta_i &= {}^{(4)}g_{0i} && \text{(2.2c)}\\ \pi^{ij} &= \sqrt{-{}^{(4)}g}({}^{(4)}\Gamma^0_{pq} - g_{pq}{}^{(4)}\Gamma^0_{rs}g^{rs})g^{ip}g^{jq} && \text{(2.2d)} \end{aligned}$$

as the relevant quantities. I call these, respectively, the spatial (three-) metric, the lapse, the shift, and the conjugate momenta. These three dimensional quantities

[2] ADM used this split in a Hamiltonian formulation of the Einstein equations, identified the lapse and shift as Lagrange multipliers enforcing the constraints, and applied the formulation to clarify the identification of total energy in asymptotically flat spacetimes.

easily relate to the spacetime metric

$$\begin{pmatrix} {}^{(4)}g_{00} & {}^{(4)}g_{0j} \\ {}^{(4)}g_{i0} & {}^{(4)}g_{ij} \end{pmatrix} = \begin{pmatrix} \beta_k\beta^k - \alpha^2 & \beta_j \\ \beta_i & g_{ij} \end{pmatrix} \tag{2.3}$$

and the inverse spacetime metric

$$\begin{pmatrix} {}^{(4)}g^{00} & {}^{(4)}g^{0j} \\ {}^{(4)}g^{i0} & {}^{(4)}g^{ij} \end{pmatrix} = \begin{pmatrix} -1/\alpha^2 & \beta^j/\alpha^2 \\ \beta^i/\alpha^2 & g^{ij} - \beta^i\beta^j/\alpha^2 \end{pmatrix} \tag{2.4}$$

where g^{ij} is the inverse of g_{ij}, and spatial indices are raised and lowered with the spatial metric. The relation $\sqrt{-{}^{(4)}g} = \alpha\sqrt{g}$, where g is the determinant of g_{ij}, is also useful.

Substituting (2.2) into (2.1) yields

$$\mathcal{L} = -g_{ij}\partial_t\pi^{ij} - \alpha H - \beta_i P^i - 2\partial_i\left(\pi^{ij}\beta_j - \frac{1}{2}\pi\beta^i + \nabla^i\alpha\sqrt{g}\right) \tag{2.5}$$

where

$$H = -\sqrt{g}\left[R + g^{-1}\left(\frac{1}{2}\pi^2 - \pi^{ij}\pi_{ij}\right)\right] \tag{2.6a}$$

$$P^i = -2\pi^{ij}{}_{;j} \tag{2.6b}$$

and $\pi = \pi^i{}_i$. Here, as throughout the text, R (with no prefix) is the Ricci scalar associated with the three-metric. The Lagrangian as expressed in (2.5) is now a function of the three dimensional quantities (2.2), and the last term is a total derivative that does not contribute to the equations of motion.

Deriving the equations of motion from this Lagrangian requires only a straightforward application of the usual tools of theoretical mechanics. The resulting evolution equations

$$\partial_t g_{ij} = 2\alpha g^{-1/2}\left(\pi_{ij} - \frac{1}{2}g_{ij}\pi\right) + \beta_{i;j} + \beta_{j;i} \tag{2.7a}$$

$$\begin{aligned}\partial_t\pi^{ij} = {} & -\alpha\sqrt{g}\left(R^{ij} - \frac{1}{2}g^{ij}R\right) + \frac{1}{2}\alpha g^{-1/2}g^{ij}\left(\pi^{mn}\pi_{mn} - \frac{1}{2}\pi^2\right) \\ & - 2\alpha g^{-1/2}\left(\pi^{in}\pi_n{}^j - \frac{1}{2}\pi\pi^{ij}\right) + \sqrt{g}\left(\nabla^i\nabla^j\alpha - g^{ij}\nabla^n\nabla_n\alpha\right) \\ & + \nabla_n\left(\pi^{ij}\beta^n\right) - \beta^i{}_{;n}\pi^{nj} - \beta^j{}_{;n}\pi^{ni}\end{aligned} \tag{2.7b}$$

come from varying the action with respect to π^{ij} and g_{ij} respectively.[3] In addition, varying with respect to α and β_i yields

$$H = 0 \tag{2.8a}$$

$$P^i = 0 \tag{2.8b}$$

which I will now call the Hamiltonian and momentum constraints.

Equations (2.7) are of fundamentally different nature than equations (2.8). This distinction is vitally important and deserves discussion. Equations (2.7) each have a single term, which I have suggestively written on the left hand sides of equations, containing a time derivative. I will call such equations *evolution equations*, because they indicate how a given degree of freedom changes with time. In contrast, equations (2.8) contain no time derivatives. I will call such equations *constraint equations* because they limit which data sets are allowed by the theory. This situation arises in all theories with gauge freedom. In the case of general relativity, the gauge freedom (the freedom to redefine coordinates) is carried in the scalar field α, which controls "how fast" time flows normal to a given spatial hypersurface, and by the vector field β^i, which controls spatial coordinate transformations undertaken between successive times. See Figure 2.1. It is a generic feature of gauge theories that variations of the action with respect to the gauge fields yield constraint equations.

Since I want to use the spacetime decomposition to formulate an initial value problem, it is important to know whether the constraints propagate. In other words, I need to answer the question: If I have data at an initial time that satisfies the constraints (2.8) and evolve that data according to the evolution equations (2.7) to a different time, will the data at the later time also satisfy the constraints? The answer is yes, the constraints do propagate. To prove this, apply the Bianchi identities $D^\mu G_{\mu\nu} = 0$, where D_μ is the covariant derivative operator compatible with the *four* dimensional spacetime metric ${}^{(4)}g_{\mu\nu}$, to the covariant Einstein tensor $G_{\mu\nu}$ of (1.1) in

[3]In order for these variations to yield meaningful results, it must be true that g_{ij} and π^{ij} are canonically conjugate variables. This follows from the form of the Hamiltonian. I have also ignored all issues related to the boundary integrals resulting from integrations by parts.

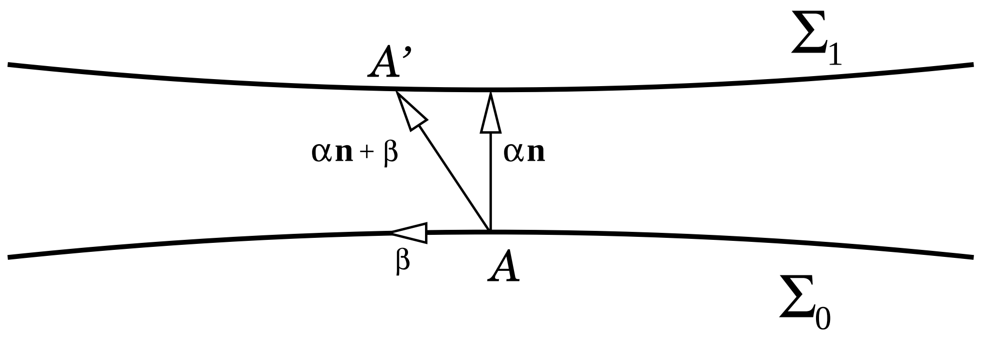


Figure 2.1: The figure demonstrates, schematically, the role of lapse and shift in the ADM decomposition. The lapse α controls how much proper time elapses between time slices Σ_0 and Σ_1 as measured along a unit normal $\mathbf{n}$. Lines of constant spatial coordinates need not run normal to the surfaces; the shift vector β^i controls the component tangent to the initial hypersurface of the vector ∂_t. In the figure the points A and A' have the same spatial coordinates (but different time coordinates).

vacuum ($T_{\mu\nu} = 0$).[4] Noting that the various projections

$$C = G_{\mu\nu} n^\mu n^\nu \tag{2.9a}$$

$$C_i = G_{\mu j} n^\mu g^j{}_i \tag{2.9b}$$

$$E_{ij} = G_{mn} g^m{}_i g^n{}_j, \tag{2.9c}$$

written in terms of the unit normal n^μ to a given foliation (as in Figure 2.1) and the spatial metric of (2.2a), correspond to the Hamiltonian constraint, momentum constraint, and the spatial portion of the Einstein tensor, respectively, the Bianchi identities projected onto the normal direction imply that

$$n^\mu D_\mu C = E_{\mu\nu} \nabla^\mu n^\nu - 2C_\nu n^\mu D_\mu n^\nu - \nabla^\mu C_\mu - C \nabla_\mu n^\mu \tag{2.10a}$$

and projected onto the spatial slice imply

$$n^\mu D_\mu C_\nu = -\nabla^\mu E_{\nu\mu} - E_{\nu\mu} n^\alpha D_\alpha n^\mu - C_\nu \nabla_\mu n^\mu - C_\mu n_\nu n^\alpha D_\alpha n^\mu - C_\mu \nabla^\mu n_\nu + 2C n^\mu D_\mu C_\nu. \tag{2.10b}$$

[4]One could also demonstrate that the constraints propagate by taking time derivatives of the constraint equations (2.8) and simplifying the resulting expressions using the evolution equations (2.7). I will follow an approach along these lines in the context of the Maxwell equations in Section 3.4. This approach is, however, more tedious for the Einstein equations.

Equations (2.10) show that if the constraints $C = C_i = 0$ are satisfied in the initial data (along with the definition $E_{ij} = 0$), then the constraints will be satisfied at all times [44].[5]

In principle, I now have a constrained initial value problem. I may choose spatial initial data for g_{ij} and π^{ij}, subject to (2.8). Once done, I am free to choose both α and β^i arbitrarily as functions of both space and time, and to use (2.7) to evolve my initial data to any later (or earlier) time. If I introduce a boundary at a finite distance, then, in addition, I will need to choose boundary conditions that also respect the constraints (2.8).

2.2 Formalisms

Equations (2.7) and (2.8) are a complete characterization of the vacuum Einstein equations in 3+1 terms. They are not, however, a *unique* representation of the Einstein equations in 3+1 terms. Notice, for example, that adding a multiple of H to the right hand side of (2.7a) does not change the physical predictions of the system of equations since all physical solutions have $H = 0$ according to (2.8a). That is not to say that the equations are the same independent of whether or not I add a multiple of the Hamiltonian constraint to some of the evolution equations, but rather that they are the same *when the constraints are satisfied.*

In a numerical simulation, the constraints will generically be violated at some level because, no matter how the simulation is designed, it always solves an approximation to the actual equations. This means that, although at the continuum level combining the constraints into a set of evolution equations as above does not effectively change the equations, any approximation to a solution is sensitive to such changes. Moreover, from numerical experiments, numerical relativists have come to understand that choosing an appropriate formalism plays a critical role in determining the long-term stability of a simulation. In current simulations, in fact, no group of which I am

[5]Note that the meaning of ∇_a and D_a are reversed in Ref. [44] relative to the conventions that I am using here.

aware uses equations (2.7) and (2.8), which I will call the "original ADM equations," as written. In the rest of this section I will introduce the formalisms that are used in this dissertation in some detail. The particular formalisms introduced in this section are by no means an exhaustive list of formalism in use today, but serve only to introduce those used in this dissertation.

2.2.1 Standard ADM

In the context of his initial data work, York [96] introduced a modification to the original ADM equations (2.7) and (2.8), which, in the numerical relativity literature, has come to be called simply the "ADM equations."[6] I follow this convention, though, when there is specific need to distinguish York's rewrite from (2.7) and (2.8), I will designate York's version of the equations "standard ADM."

Rather than evolving the canonically conjugate momenta π^{ij}, York chose the extrinsic curvature K_{ij} of the three dimensional slices as embedded in the four dimensional spacetime manifold. In pure geometrical terms, the extrinsic curvature is a measure of the "bend" of a hypersurface as measured from a higher dimensional space in which hypersurface is embedded. For my purposes here, it is sufficient to note that

$$K_{ij} = -g^{-1/2} \left(\pi_{ij} - g_{ij} \pi \right) \tag{2.11}$$

gives a simple algebraic relationship between π^{ij} and K_{ij}.

In this formalism, the evolution equation for the three-metric g_{ij} is (2.7a), using (2.11) to eliminate π_{ij} in favor of the extrinsic curvature K_{ij}. The evolution equation for the extrinsic curvature, on the other hand, comes from taking a time derivative of (2.11), and replacing the time derivatives that appear on the right side of the equations according to (2.7b) and the new evolution equation for the metric (2.12a). In passing from the original ADM equations to the standard equations, it is also

[6]York was not the first to write the equations in form described in Section 2.2.1. Indeed the original ADM paper, Ref. [9], discusses the possibility of using the extrinsic curvature as the "momentum" variable. York's work, however, appears to be what made that choice popular in the numerical relativity community.

necessary to solve the Hamiltonian constraint (2.8a) for the Ricci scalar R in terms of g_{ij} and K_{ij}, and use this expression for R to remove the Ricci scalar from the evolution equations. (York provided a different derivation, and from the point of view that he adopted in Ref. [96], the equations below follow naturally, without the need to explicitly incorporate the Hamiltonian constraint.) The final results

$$\partial_t g_{ij} = -2\alpha K_{ij} + \beta_{i;j} + \beta_{j;i} \tag{2.12a}$$

$$\partial_t K_{ij} = -\nabla_i \nabla_j \alpha + \alpha \left(R_{ij} - 2K_{in}K^n_{\ j} + KK_{ij} \right) + \pounds_\beta K_{ij} \tag{2.12b}$$

take this mixing of the constraint into the evolution equation into account, with

$$H = R + K^2 - K_{ij}K^{ij} \tag{2.13a}$$

$$P^i = \left(g^{im}g^{jn} - g^{ij}g^{mn} \right) K_{mn;j} \tag{2.13b}$$

giving the constraints themselves.[7]

2.2.2 BSSN

An alternative to ADM was suggested in a paper by Shibata and Nakamura in 1995 [77] and was made popular by a subsequent paper by Baumgarte and Shapiro in 1999 [12]. It has since become known as the BSSN formalism, and it has all but replaced ADM in modern simulations because, empirically, it has better stability properties.

Discussion begins with the definition of new variables

$$\phi = \frac{1}{12} \log g \tag{2.14a}$$

$$K = g^{ab} K_{ab} \tag{2.14b}$$

$$\tilde{g}_{ij} = e^{-4\phi} g_{ij} \tag{2.14c}$$

$$\tilde{A}_{ij} = e^{-4\phi} \left(K_{ij} - \frac{1}{3} g_{ij} K \right) \tag{2.14d}$$

$$\tilde{\Gamma}^i = \tilde{g}^{ab} \tilde{\Gamma}^i_{ab} \tag{2.14e}$$

in terms of the ADM variables. In this formalism, it is the quantities on the left hand sides of equations (2.14) that are independently evolved. This means that

[7]Equations (2.13) are rescaled by a factor of $g^{-1/2}$ relative to equations (2.8). Since their analytic value is zero, this is physically meaningless, but it is nonetheless convenient.

there are seventeen free variables in the system, compared to twelve free variables in the ADM equations.[8] This, however, is no problem because equations (2.14) provide five constraints on the BSSN variables, which, in addition to the physical constraints (2.8), should be satisfied by the data. More specifically, the five constraints are as follows: (2.14a) and (2.14c) imply that $\det \tilde{g} = 1$; (2.14b) and (2.14d) imply that $\tilde{g}^{ij}\tilde{A}_{ij} = 0$; and (2.14e) itself is three constraints between $\tilde{\Gamma}^i$ and the conformal metric $\tilde{g}_{ij}$.

The evolution equations for these variables are derived from the evolution equations for the ADM variables (2.12). In computing the time derivative for $\tilde{\Gamma}^i$, which is typically called the *conformal connection*, it is again necessary to mix the constraints into the evolution equations in order to have a stable evolution system. It is conventional to use the momentum constraint (2.8b) to replace the $\tilde{A}^{ij}{}_{,j}$ where it appears in the evolution equation for $\tilde{\Gamma}^i$ because it has been found empirically that it is essential in making this formalism stable [12, 5]. It has also been argued on purely mathematical grounds that adding the momentum constraint to this evolution equation plays a key role in making the BSSN system numerically stable [67].

In the end

$$\begin{aligned}
\partial_t \phi &= -\frac{1}{6}\alpha K + \pounds_\beta \phi & (2.15\text{a})\\
\partial_t K &= -\nabla^a \nabla_a \alpha + \alpha\left(\tilde{A}^{ab}\tilde{A}_{ab} + \frac{1}{3}K^2\right) + \pounds_\beta K & (2.15\text{b})\\
\partial_t \tilde{g}_{ij} &= -2\alpha \tilde{A}_{ij} + \pounds_\beta g_{ij} & (2.15\text{c})\\
\partial_t \tilde{A}_{ij} &= e^{-4\phi}\left(-\nabla_i \nabla_j \alpha + \alpha R_{ij}\right)^{\mathrm{TF}} + \pounds_\beta \tilde{A}_{ij} & \\
&\quad + \alpha\left(K\tilde{A}_{ij} - 2\tilde{A}_{ia}\tilde{A}^a{}_j\right) & (2.15\text{d})\\
\partial_t \tilde{\Gamma}^i &= 2\alpha\left(\tilde{\Gamma}^i_{ab}\tilde{A}^{ab} - \frac{2}{3}\tilde{g}^{ia}K_{,a} + 6\tilde{A}^{ia}\phi_{,a}\right) & \\
&\quad + \tilde{g}^{kl}\left(-\tilde{\Gamma}^j_{kl}\beta^i{}_{,j} + \frac{2}{3}\tilde{\Gamma}^i_{kl}\beta^j{}_{,j}\right) + \beta^k\tilde{\Gamma}^i{}_{,k} & \\
&\quad + \tilde{g}^{jk}\beta^i{}_{,jk} + \frac{1}{3}\tilde{g}^{ij}\beta^k{}_{,kj} - 2\tilde{A}^{ia}\alpha_{,a} & (2.15\text{e})
\end{aligned}$$

[8]This counting takes into account that the 3×3 matrices g_{ij}, K_{ij}, $\tilde{g}_{ij}$, and $\tilde{A}_{ij}$ are symmetric, and therefore have only six independent components each.

are the evolution equations. Equations (2.15) require more explanation. First, the "TF" in (2.15d) indicates the trace-free part of the expression in the parentheses. Second, note that the covariant derivatives in (2.15b) and (2.15d) are with respect to the *physical* metric g_{ij}. Although convenient for writing the equations, in code this should be computed in terms of BSSN quantities. The full expression is

$$\nabla_m \nabla_n \alpha = \partial_m \partial_n \alpha - 4\partial_{(m}\phi\partial_{n)}\alpha - \tilde{\Gamma}^k_{mn}\partial_k\alpha + 2\tilde{\gamma}_{mn}\tilde{\gamma}^{kl}\partial_k\phi\partial_l\alpha \tag{2.16}$$

(see, for example, [55]). The index should be raised in (2.15b) with the physical metric g^{ij}. Third, the Ricci tensor in (2.15d) is also with respect to the physical metric. It can be computed according to the decomposition

$$R_{ij} = \tilde{R}_{ij} + R^{\phi}_{ij} \tag{2.17}$$

with

$$\begin{aligned} \tilde{R}_{ij} &= -\frac{1}{2}\tilde{g}^{lm}\tilde{g}_{ij,lm} + \tilde{g}_{k(i}\tilde{\Gamma}^k{}_{,j)} + \tilde{g}^{lm}\tilde{\Gamma}^k_{lm}\tilde{\Gamma}_{(ij)k} \\ &\quad + \tilde{g}^{lm}\left(2\tilde{\Gamma}^k_{l(i}\tilde{\Gamma}_{j)km} + \tilde{\Gamma}^k_{im}\tilde{\Gamma}_{klj}\right) \end{aligned} \tag{2.18}$$

giving the Ricci tensor associated with the conformal metric, and

$$R^{\phi}_{ij} = -2\tilde{\nabla}_i\tilde{\nabla}_j\phi - 2\tilde{g}_{ij}\tilde{\nabla}^k\tilde{\nabla}_k\phi + 4\tilde{\nabla}_i\phi\tilde{\nabla}_j\phi - 4\tilde{g}_{ij}\tilde{\nabla}^k\phi\tilde{\nabla}_k\phi \tag{2.19}$$

giving the part dependent on the conformal factor.

As written, equations (2.15) and (2.18) also reflect the empirically correct choices for when to use the independently evolved $\tilde{\Gamma}^i$, and when to recompute the equivalent quantity from the conformal metric.[9] Whenever derivatives of the conformal connection appear, the independently evolved quantity is used, but where the conformal connection itself would appear, it is recomputed from the conformal metric according to (2.14e). This has been shown empirically to lead the most stable numerical system [5].

[9]The community still experiments with this point. As written here, I have followed the general rules laid out in Ref. [4] and used in work related to this dissertation, Ref. [55].

Chapter 3

Constraints

Constraints are ubiquitous in classical physics. They appear in even fairly elementary problems of dynamics, and appear again, in a slightly different way, in gauge field theories. This chapter talks briefly about the nature of constraints in physical theories at a continuum level, before reanalyzing them in the context of numerical simulations. The behavior of systems in the two cases is radically different, in that constraints, which at the continuum level are forever satisfied if satisfied in the initial data, may grow when the equations are discretized. After these two short introductory sections, I present an original method for controlling the growth of constraints in numerical simulations by incorporating constraint equations into evolution equations in a novel way.

Before proceeding to a more detailed discussion, I find it useful to define, and to distinguish the difference between, two terms.

Definition 3.1 *A* constraint $C = C(t, \mathbf{x})$ *is a function of space and time such that* $C(t, \mathbf{x}) = 0$ *for all* t *and* $\mathbf{x}$.

Definition 3.2 *A* conserved quantity $C = C(t)$ *is a function of time only such that* $C(t) = 0$ *for all* t.

A few comments are needed. First, it is immediately clear from the definitions that a constraint always implies a conserved quantity, but that the reverse is not true. Second, note that, without loss of generality, I define constraints to be zero-valued. This is convenient for technical reasons that will be apparent later. Finally, notice that conserved quantities are frequently expressed as integrals. Total mass, for example, is conserved in an integral sense meaning that mass density at any given point may changed over time, but the integral over the whole space is constant. Constraints, on the other hand, are usually specified as functions and are identically satisfied pointwise throughout the space. This chapter is concerned only with constraints as defined here. The constraint driver methods, in particular, introduced in Section 3.3, are not generally applicable to conserved quantities.

3.1 Constraints in Hamiltonian systems

In classical mechanics, constraints usually arise to account for forces that are not otherwise treated explicitly. A roller-coaster car, which is confined to its track, is an obvious example. The problem can be solved, without worrying about the details of forces that hold the car to the track, by treating the path of the track as a constraint in the problem. The constraint, in such a case, essentially indicates that, while we may think of the car moving in three dimensions, the mathematical problem is only one dimensional. In the case of holonomic constraints,[1] the typical method for solving such a problem in N dimensions with C constraints is to introduce $N - C$ generalized coordinates, which implicitly include the constraints. In effect, switching to generalized coordinates makes the actual number of degrees of freedom in the problem manifest.

Constraints arise again in gauge theories, though the way that they enter the theory is different. Here the physical fields of the theory are expressible in terms of other, non-observable fields. The Maxwell equations, in which the scalar and vector potentials are gauge fields, are the simplest example in physics. Changing the gauge fields in a way consistent with the theory does not alter the physical fields in any

[1] *Holonomic* constraints are those that can be expressed as an *equality* between coordinates of particles. The roller-coaster car is an example. *Nonholonomic* constraints are all others. The walls of a container filled with a gas are an example. See Goldstein [46] for more discussion on this point in classical mechanics.

way. The precise physical meaning of the constraints in a gauge theory is not as clear as it was in typical case of classical mechanics. In a mathematical sense, they again indicate that the physical fields have "too many" degrees of freedom, but the physical origin of this excess remains a much deeper question in general.[2] Whatever the cause, however, the fact remains that the physical fields of a gauge theory are necessarily subject to constraints.

In the context of solving a gauge theory's equations of motion as an initial value problem, a natural question arises: Is it necessary to solve the constraints at each moment of time, or is it sufficient to solve them only in initial data? The answer, for a self-consistent theory, is that it is sufficient to solve the constraints in the initial data because the constraints of the theory propagate. This is generally proved by computing the time derivative of a constraint, and showing, by substituting the evolution equations for the fundamental fields into this equation, that the evolution equation for the constraint is a linear combination of the constraints in the theory. Given that the constraints are satisfied at the initial time, this proves that they remain satisfied for all time.[3]

One might also ask, when trying to solve the equations of motion for gauge field theory, if some analogue of generalized coordinates exist in which the true degrees of freedom of the theory are isolated, eliminating the need to further deal with the constraints. The answer here is mixed. For the Einstein equations, which are of primary concern here, the answer appears to be that one cannot remove the constraints entirely. The momentum constraint can be removed from the theory by considering only equivalence classes of three-metrics, where any two three-metrics related by a diffeomorphism are considered equivalent. This leaves, however, the

[2]In the case of Maxwell, the gauge freedom is related to electric charge conservation and the lack of magnetic monopoles. In the case of Einstein, it is related to coordinate freedom. Why there should be charge conservation or coordinate freedom, however, is a fundamentally deeper question than the purely mechanical question of why the car should stay on the track.

[3]See Appendix E of Wald [89] for extended discussion on this point in terms of the Maxwell, the Klein-Gordan, and the Einstein equations. See Section 2.1 for an alternate approach for the case of the Einstein equations.

Hamiltonian constraint. As of early 2004, it does not appear that this constraint can be removed by any such trick, in large part because it is quadratic in the fields and therefore does not admit a unique inversion. In addition to posing problems for numerical relativists, this also presents a difficulty for those attempting canonical quantization of gravity [89].

3.2 Constraints in numerical simulations

Although there are analytic guarantees that, if the initial data satisfies the constraints, and that initial data is propagated forward by the exact evolution equations, the constraints will be satisfied at all future times, numerically this need not be true. Round-off and truncation errors necessarily introduced in the numerical approximation to the problem will take the simulated solution off of the constraint hypersurface. Once off of the constraint hypersurface, even analytically, there are no guarantees about how the system will behave since this is a non-physical region of the solution space.

In practice, there are three broad classes of techniques used to deal with constraints in a numerical evolution: (1) *Free evolution*, in which the initial data is constructed to satisfy the constraints, and is evolved forward using the evolution equations. The constraint violation introduced by numerical error is monitored, but no action is taken to re-solve the constraints at later times. (2) *Special numerics*, by which I mean numerical schemes that are tailored to a specific problem and are designed to ensure that the constraints are satisfied up to some order. (3) *Constrained evolution*, in which the constraints are solved explicitly throughout the evolution, either by taking the evolved solution as a starting point for an elliptic solver, or by making algebraic substitutions on the evolved solutions such that the modified data satisfies the constraints. For the purposes of this dissertation, only free evolution has direct relevance. Attempts to use fully constrained evolution have so far been extremely limited in full, three dimensional numerical relativity, and, to date, no one has successfully formulated and implemented the Einstein equations in a way that admits special numerics that identically conserve constraints.

There are two primary reasons, that constrained evolution techniques have not been popular, although they have been and are practiced by some in the field [73, 7]. The first is that the constraint equations of the Einstein system are elliptic equations, and finding numerical solutions to elliptic equations tends to be quite computationally expensive. The second, more fundamental reason is that the procedure may not be well defined. Solutions to elliptic equations are entirely determined by boundary conditions, whereas no one knows how to sensibly apply a boundary condition near a black hole. Even if the physical boundary of a computational domain is far enough from a black hole to apply a reasonable boundary condition there, most long-term stable simulations include excision techniques (i.e. they do not evolve data in the interior of the black hole — see, for example, [3, 79]), in which a boundary is introduced inside the event horizon of the black hole. Specifying an appropriate boundary condition here is an open question for the elliptic problem.

In order to avoid these problems, there have been some numerical experiments with solving the constraints algebraically. Instead of solving the elliptic equations explicitly, there have been some attempts to substitute constraints for evolution equations to ensure that the constraints are satisfied (e.g. Ref. [41]). These methods have not become as popular as free evolution methods, nor are they entirely natural since it is not clear how to substitute four constraint equations into twelve (first order in time) evolution equations in a meaningful way.[4]

Given that constraint violating modes can be created by the numerics used in a simulation, that constraint violating modes have empirically been linked to numerical instabilities, and that the dynamics of the fields off of the constraint hypersurface are not constrained by physics, it is not surprising that a variety of authors (e.g. [9, 77, 12, 5, 56, 59, 93]) have found that exact form in which the evolution equations are written plays a role in the stability of simulations. Exactly how to rewrite the

[4]There is a notable exception in the BSSN formalism. It is common to enforce the constraint $\tilde{g}^{ij}\tilde{A}_{ij} = 0$ by subtracting the trace of $\tilde{A}_{ij}$ from the evolved variable at the end of each iteration of the time stepping algorithm. See, for example, Ref. [55].

evolution equations in order to increase stability has, however, remained more of an art than a science.

3.3 Driver terms

Making one trivial observation can lead to an unlimited number of ways of dealing with the constraints of the system. Given the evolution equations and constraint equations written in a particular formalism, one can freely modify the evolution equations by adding multiples of the constraints. Analytically this does not change the physics of the system because, on the constraint hypersurface, these terms vanish identically. Off of the constraint hypersurface, these terms will be non-zero, and will therefore affect the off-constraint dynamics of the evolution system under free evolution. Choosing the exact form of these corrections carefully might induce the evolution to dynamically favor constraint satisfying solutions. This observation, in many variations, has been applied, to differing degrees of success, by other authors [37, 21, 94, 78, 95, 88, 7, 60].

In what follows, I repeat and expand upon results that I first reported in Ref. [42]. My goal in this work is to develop a general prescription for choosing such terms. This approach differs significantly from other such attempts in that it (1) considers non-linear correction terms, (2) provides a general analytic prescription for generating the correction terms, or (3) both. In order to present the idea, I first discuss the ordinary differential equation case of a simple harmonic oscillator. In Section 3.4, I turn to how to treat partial differential equations with an extensive application to the Maxwell equations, written in two different formalism, each of which closely resemble popular formalisms of the Einstein equations. Finally, in Section 3.5 I study the linearized Einstein equations in one spatial dimension.

3.3.1 Simple Harmonic Oscillator

The equations of motion for the simple harmonic oscillator, written in first order form are

$$\frac{d}{dt}\begin{pmatrix} x \\ v \end{pmatrix} = \begin{pmatrix} v \\ -x \end{pmatrix} \tag{3.1}$$

in units where the angular speed $\omega = 1$. For a constraint, take the energy of the system at time t shifted by the initial energy $C(t) = x^2(t) + v^2(t) - E_0$, which, analytically, should be zero for all t.

Given that the constraint may be violated in a numeric integration, I would like a way to measure whether the system will tend to evolve toward or away from the proper constraint value. In making this determination, it is useful to consider the squared constraint since this is of definite sign. Consider the time evolution of the squared constraint. By the chain rule,

$$\frac{dC^2}{dt} = \frac{\partial C^2}{\partial x}\frac{dx}{dt} + \frac{\partial C^2}{\partial v}\frac{dv}{dt} \tag{3.2}$$

gives this time evolution. Notice now that the functional form of the partial derivatives in (3.2) is fixed by the form of the energy expression of this formalism, but the time derivative terms on the right hand side of (3.2) may be modified by adding multiples of the constraints without changing the physics of the system.

Consider the modified system of evolution equations

$$\frac{d}{dt}\begin{pmatrix} x \\ v \end{pmatrix} = \begin{pmatrix} v \\ -x \end{pmatrix} - K\begin{pmatrix} \partial_x C^2 \\ \partial_v C^2 \end{pmatrix} \tag{3.3}$$

with K a positive constant. Repeating the chain rule argument outlined above,

$$\frac{dC^2}{dt} = \left[\frac{\partial C^2}{\partial x}v - \frac{\partial C^2}{\partial v}x\right] - K\left[\left(\frac{\partial C^2}{\partial x}\right)^2 + \left(\frac{\partial C^2}{\partial v}\right)^2\right] \tag{3.4}$$

gives the new evolution equation for the constraint. The first term in brackets is a piece determined by the underlying formalism, while the second term is due entirely to the correction applied in (3.3). Moreover, by the choice of K, the correction term has definite sign and will tend to restore the system to a state of zero constraint.

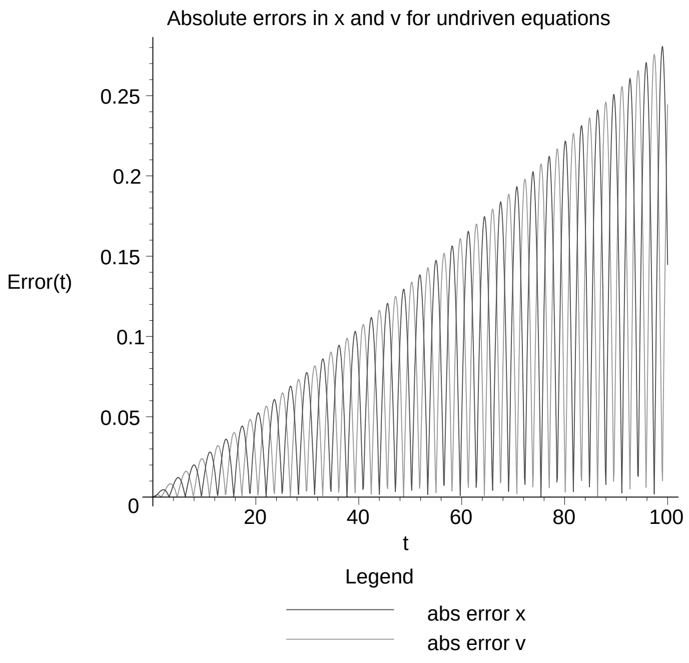


Figure 3.1: The time evolution of the solution error in x and v for the undriven ($K = 0$) evolution equations.

This effect is balanced against the unknown sign of the first term, so that success is not guaranteed, but by choosing K large enough one might hope that the dynamics would choose to stay near the constraint hypersurface.

Numerical simulations, in fact, bear out these hopes. I ran a toy model based on (3.3) in Maple [62], testing the system with different values of K. I integrated forward in time with the Euler method and looked at both the solution error and the constraint violation as a function of time.

The solution error results are displayed in Figures 3.1–3.3. Notice that while solution error grows linearly with time in the unmodified system ($K = 0$), as seen in Figure 3.1, it quickly levels out to a relatively small value in Figure 3.2 ($K = 1$) and grows linearly with a slope 3 orders of magnitude smaller than the unmodified case when $K = 100$ in Figure 3.3.

Likewise, the results for the constraint violation itself is equally encouraging. Compare the constraint violation for the unmodified equations, shown in Figure 3.4, which grows for the entire duration of the simulation, to the constraint violation in

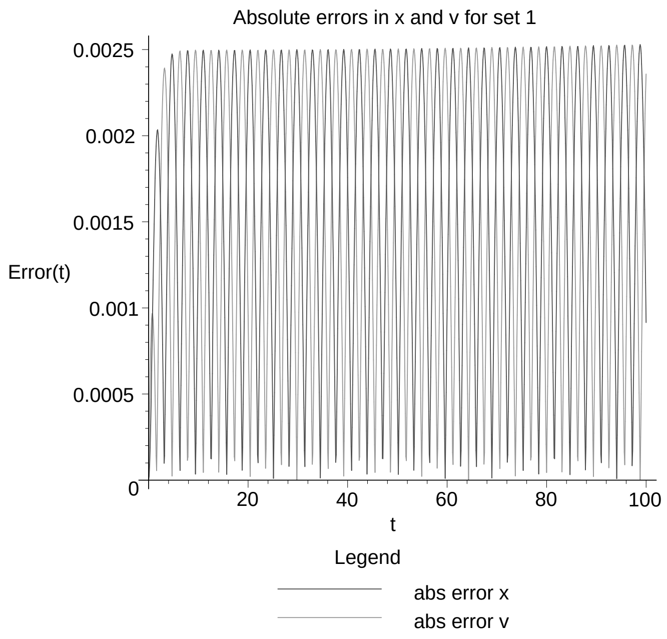


Figure 3.2: The time evolution of the solution error in x and v for the driven evolution equations with $K = 1$.

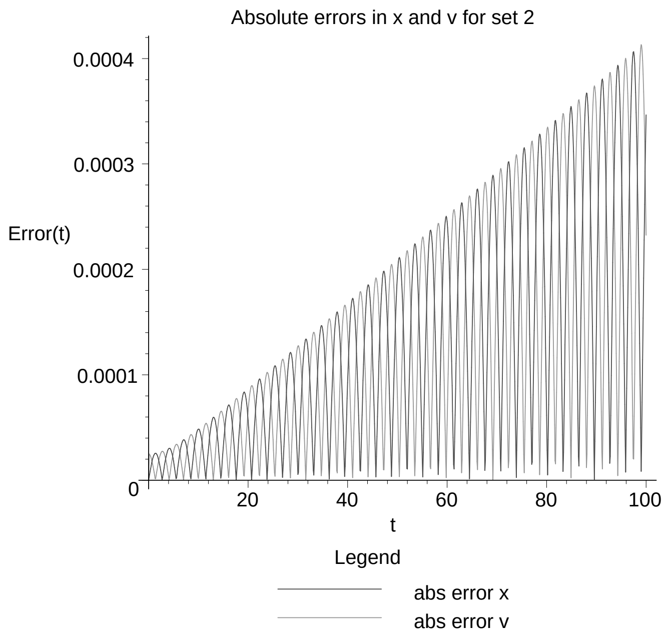


Figure 3.3: The time evolution of the solution error in x and v for the driven evolution equations with $K = 100$.

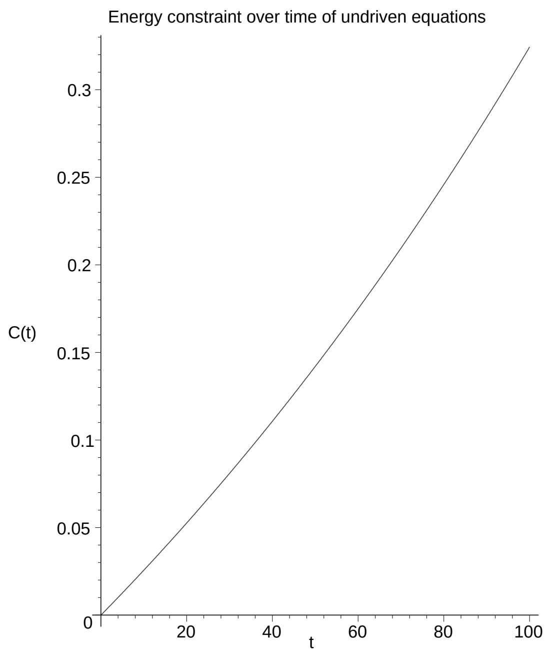


Figure 3.4: The time evolution of the constraint $C(t)$ for the undriven ($K = 0$) evolution equations.

the two modified runs, shown in Figure 3.5 and Figure 3.6 ($K = 1$ and $K = 100$, respectively). In both modified cases, the constraint violation quickly levels out, staying at a fixed value for the duration of the run. This fixed value, as expected, is smaller for the larger value of K.

By putting the spring constant k and the mass m back into the harmonic oscillator equations, and then making the spring constant a function of the energy of the system, I was able to construct a more complicated toy model, where the off-constraint behavior depended on the deviation from the proper constraint value. The results were substantially the same, with both the solution error and the constraint violation larger in the unmodified system of equations. The modified equations performed better with larger values of K than with smaller values.

3.3.2 Partial Differential Equations

Having seen the method in the simple case of ODEs, consider now the more interesting case of PDEs with one constraint equation. Let s be the state vector for a

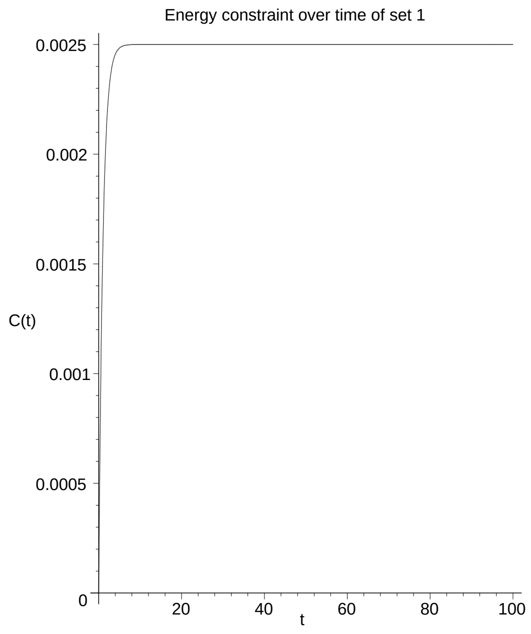


Figure 3.5: The time evolution of $C(t)$ for the driven evolution equations with $K = 1$.

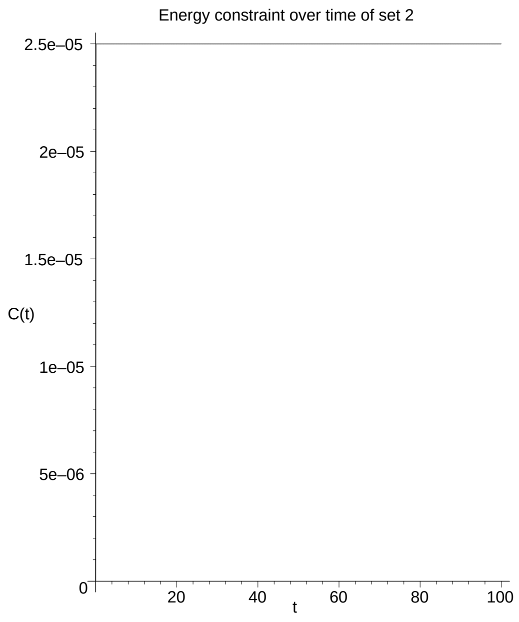


Figure 3.6: The time evolution of $C(t)$ for the driven evolution equations with $K = 100$.

system. Define

$$S_m(t,\mathbf{x}) = \frac{\partial s_m}{\partial t} \tag{3.5}$$

to be the right hand side of the evolution equation in the unmodified formalism. A general constraint C will depend on s and its spatial derivatives. Furthermore, the constraint should be satisfied at every point in space. These considerations motivate looking at the integrated, squared constraint $\overline{C^2} = \int C^2 d^N x$, and, instead of taking partial derivatives with respect to the fields, I need to take variational derivatives of the integrated constraint when considering the analogies of (3.3) so that the dependence of C on the spatial derivatives of the fields is treated properly.

Following this prescription, the appropriate modification to the equation of motion for the state vector is

$$\frac{\partial s_m}{\partial t} = S_m(t,\mathbf{x}) - K_{mn}(t,\mathbf{x})\frac{\delta\overline{C^2}}{\delta s_n(t,\mathbf{x})} \tag{3.6}$$

for some positive-definite matrix-valued function K_{mn}. Under this change,

$$\frac{d\overline{C^2}}{dt} = D[s] - \int \left(\frac{\delta\overline{C^2}}{\delta s_m}\right) K_{mn} \left(\frac{\delta\overline{C^2}}{\delta s_n}\right) d^N x \tag{3.7}$$

gives the evolution of the constraint in the modified theory. Here $D[s]$ gives the functional form of the right hand side of the constraint's evolution equation in the unmodified theory. For the cases considered here, I chose the K_{mn} diagonal and constant.

For systems with M constraint equations, the method is easily modified by taking the grand constraint functional to be

$$\overline{C_G^2} = \int w_{IJ}(t,\mathbf{x}) C_I C_J d^N x \tag{3.8}$$

with any positive definite matrix w_{IJ}. Like K_{mn}, the matrix w_{IJ}, can, in principle, be a function of both space and time, and can have off-diagonal entries. In practice, however, I have only used diagonal and constant matrices. Even in the diagonal case, the matrix is necessary in principle because there is no a priori reason to believe that the constraints have the same measurement dimensions. It also allows the different constraints to be treated with different relative importance. In addition, since there is no natural scale for the grand constraint, one may always set one of the coefficients w_{IJ} in (3.8) to unity.

3.4 Application to the Maxwell Equations

This section takes Maxwell's equations as a concrete example of a system of partial differential equations subject to a pointwise constraint. Knapp, Walker, and Baumgarte (KWB) [58] showed both that the Maxwell equations may be formulated in ways closely paralleling popular formulations of the Einstein equations, and that numerical properties of the full Einstein equations can be understood more easily in the simpler context of the Maxwell equations. Following KWB, I consider two ways of writing the vacuum Maxwell equations in terms of the vector potential A_i. The first system, called System I, uses the evolution equations

$$\begin{aligned} \partial_t A_i &= -E_i - \partial_i \psi && (3.9) \\ \partial_t E_i &= -\partial_j \partial_j A_i + \partial_i \partial_j A_j && (3.10) \end{aligned}$$

and the constraint

$$C_E = \partial_i E_i = 0. \quad (3.11)$$

The second system introduces the additional field Γ defined by

$$\Gamma = \partial_i A_i \quad (3.12)$$

to eliminate mixed derivatives in (3.10). The evolution equations for System II are

$$\begin{aligned} \partial_t E_i &= -\partial_j \partial_j A_i + \partial_i \Gamma && (3.13) \\ \partial_t \Gamma &= -\partial_i \partial_i \psi && (3.14) \end{aligned}$$

and (3.9). Both systems use a gauge consistent with

$$\partial_t \psi = -\partial_i A_i = -\Gamma \quad (3.15)$$

using the first equality for System I and the second for System II.

3.4.1 System I Evolution Equations

Having defined the systems, I would now like to calculate the terms required for applying the constraint finding method to System I. Here there is only one constraint, $C = C_E = \partial_i E_i$, which is zero-valued. It depends only on the first derivatives of the electric field, therefore I need only to calculate

$$\frac{\delta \overline{C^2}}{\delta E_i(t, \mathbf{x})} = -2\partial_i C_E \tag{3.16}$$

which modifies (3.10). The new evolution equation for the electric field is

$$\partial_t E_i = -\partial_k \partial_k A_i + \partial_i \partial_k A_k + 2K_E \partial_i C_E \tag{3.17}$$

for an arbitrary positive K_E, while the other System I evolution equations (3.9) and (3.15) remain unchanged.

3.4.2 System II Evolution Equations

System II, unlike System I, has two constraints that should be enforced, the original constraint given by (3.11) plus the definition of Γ in (3.12), rewritten as

$$C_\Gamma = \partial_i A_i - \Gamma = 0 \tag{3.18}$$

to make it zero-valued. This provides more freedom in constructing the grand constraint

$$C^2 = C_E^2 + wC_\Gamma^2 \tag{3.19}$$

where one does not necessarily have to treat the constraints on equal footing. In this case, the total constraint depends additionally on Γ and on first derivatives of the vector potential. In addition to (3.16), which is still valid, I need

$$\frac{\delta \overline{C^2}}{\delta A_i(t, \mathbf{x})} = -2w\partial_i C_\Gamma \tag{3.20}$$

$$\frac{\delta \overline{C^2}}{\delta \Gamma(t, \mathbf{x})} = -2wC_\Gamma \tag{3.21}$$

to enforce the definition of Γ.

Applying these correction terms to the evolution equations gives the new equations of motion

$$\begin{aligned}
\partial_t A_i &= -E_i - \partial_i \psi + 2wK_A \partial_i C_\Gamma && (3.22)\\
\partial_t E_i &= -\partial_k \partial_k A_i + \partial_i \Gamma + 2K_E \partial_i C_E && (3.23)\\
\partial_t \Gamma &= -\partial_k \partial_k \psi + 2wK_\Gamma C_\Gamma && (3.24)
\end{aligned}$$

which, combined with (3.15), form a complete system. The constants K_A and K_Γ are arbitrary but positive.

3.4.3 Propagation of Constraints

In Ref. [58], KWB examined the evolution equation for C_E in both systems. They showed that for System I, the constraint does not evolve in time, and that in System II, the constraint obeys a wave equation. They did not have reason, however, to consider the time evolution of the secondary constraint C_Γ. I extend their results here by showing that the secondary constraint also satisfies a wave equation in the unmodified case. Furthermore, I demonstrate the improved behavior of the constraints under the modifications that I have proposed.

Calculating the first time derivatives of the constraints is easily accomplished by taking the time derivatives of (3.11) and (3.18) and replacing the time derivatives that appear on the right hand sides of the equations by the evolution equations (3.17), (3.22), and (3.24). This gives the results

$$\begin{aligned}
\partial_t C_E &= -\partial_i^2 \left[C_\Gamma - 2K_E C_E \right] && (3.25a)\\
\partial_t C_\Gamma &= -C_E + 2w \left[K_A \partial_i^2 C_\Gamma - K_\Gamma C_\Gamma \right] \; . && (3.25b)
\end{aligned}$$

Equation (3.25a) can be viewed as valid for both systems if C_Γ is taken to be identically zero for System I.

To see that KWB have a wave equation for the secondary constraint, set all of the Ks to zero in (3.25) to eliminate the modifications, and take the time derivative of (3.25b). The result

$$\partial_t^2 C_\Gamma = \partial_i^2 C_\Gamma \qquad (3.26)$$

follows immediately.

Of greater interest here, however, is an analysis on the modified equations (3.25) in their first derivative form. The equations are linear, so they admit a Fourier analysis by substituting a plane wave solution e^{ikx} into the right hand sides. After this substitution, the resulting equations

$$\partial_t C_E = k^2 C_\Gamma - 2K_E k^2 C_E \tag{3.27a}$$

$$\partial_t C_\Gamma = -C_E - 2w\left[K_A k^2 + K_\Gamma\right] C_\Gamma \tag{3.27b}$$

retain the terms that gave the KWB wave equations for the constraints, but have additional terms that look like they provide exponential decay. This system of equations is, in fact, simple enough for Maple to solve analytically for general values of the Ks in one dimension. The solution, which is too long to display in detail here, consists of a sum of terms with the form

$$\exp\left[\left(-f_1^+ \pm \sqrt{\sigma\left[\left(f_1^-\right)^2 - k^2\right]}\right) t\right] f_2 \tag{3.28}$$

where

$$f_1^\pm(k^2) = (K_E \pm wK_A)k^2 \pm wK_\Gamma \tag{3.29}$$

$\sigma = \pm 1$, and $f_2 = f_2(k, C_i(0,x))$ is some simple function of k and the initial values of the constraints. Since $f_1^+(k^2) > 0$ is manifestly positive and the radical is either positive or pure imaginary, the only term of this form that could cause anything other than exponential decay is $-f_1^+ + \sqrt{\sigma[(f_1^-)^2 - k^2]}$ for parameters where $\sigma[(f_1^-)^2 - k^2] > 0$. Simple algebraic analysis, however, shows that even this term has an overall minus sign, giving exponential decay. In order to make the argument more concrete, I present the $k = 1$ solution

$$C_E(t,x) = e^{-3t}\left[C_E(0,x) + S(x)t\right] \tag{3.30a}$$

$$C_\Gamma(t,x) = e^{-3t}\left[C_\Gamma(0,x) - S(x)t\right] \tag{3.30b}$$

for which all of the Ks are set equal to one. Here $S(x) = C_E(0,x) + C_\Gamma(0,x)$ is a short hand.

Run	K_E	K_A	K_Γ	w
I-0	0	-	-	-
I-1	1×10^{-2}	-	-	-
I-2	5×10^{-2}	-	-	-
II-0	0	0	0	1
II-1	5×10^{-3}	5×10^{-3}	5×10^{-3}	1
II-2	1×10^{-2}	1×10^{-2}	1×10^{-2}	1

Table 3.1: The parameters used for the various simulations of the two Maxwell systems are tabulated here.

One might object that this constraint behavior is "too good" (as did Tiglio [88]). This result seems to indicate that the constraint violations, at late times, will be pushed below truncation error, which seems unnatural if not unstable for a numerical calculation. In practice, as I will show with numerical experiments, truncation errors prevent complete exponential decay in the constraints (as they must), but the continuum tendency toward exponential decay keeps the constraints well controlled.

3.4.4 Numerical Results

My numerical experiments on these modified systems of equations used an ICN integration scheme [87], and a Courant factor of 1/2. The spatial domain ran from -6 to $+6$, with data stored on 99 points in each coordinate direction. On the evolved fields (E_i, A_i, ψ, and Γ), I imposed outgoing wave boundary conditions (see Ref. [2] for implementation details), and, on the constraints, I imposed $C_I = 0$ for applicable I. All runs were performed on a 500 MHz Digital Personal Workstation with 1.5 GB of RAM.

For each system, I ran three parameter sets, one of which reproduced the equations used by KWB. The full definitions of all of the parameter sets are found in Table 3.1. Sensible values of the parameters were easily determined by trial and error. Choosing the values too small, as expected, makes little difference in the evolution, while choosing the values too large leads to numerical instabilities during the transient period of the evolution.

In my original paper on this subject, Ref. [42], I noted that one possible explanation for the instability associated with large parameters is that the added terms modify the dispersion relationships for the various Fourier modes, as seen in (3.28). Large values of the parameters may require adjustments to the Courant condition, which I had not analyzed in detail. In a later paper, Calabrese [26], in fact, made this statement more precise. The correction terms in the case of the Maxwell equations change the system from a hyperbolic system to a mixed parabolic-hyperbolic system. While for the hyperbolic system the standard Courant condition $dt = \lambda dx$ with $\lambda < 1$ applies, in the case of parabolic equations stability requires $dt = \lambda dx^2$, where λ depends on the size of the damping parameters. I address this issue further in Section 3.5 in the context of the Einstein equations.

I followed KWB in using the analytic solution

$$A^\phi = 0 \tag{3.31a}$$

$$E^\phi = 8\mathcal{A}\lambda^2 r e^{-\lambda r^2} \sin\theta \tag{3.31b}$$

of a toroidal dipole to generate the initial data. The other components of the fields are zero. I chose $\lambda = \mathcal{A} = 1$, and the conversion from spherical to Cartesian coordinates was made in the code.

The results of the System I runs are summarized in Figure 3.7, which shows a plot of $\|C_E\|_2$ versus t. The I-0 (control) curve reproduces the findings of KWB that, after an initial transient, the C_E constraint does not evolve in time. The I-1 and I-2 curves, representing different values (see Table 3.1) of the parameter K_E, on the other hand, show a modulated exponential decay. Eventually, around $t \approx 200$, the I-1 case also stops decaying as rapidly, while the rapid exponential decay continues through the end of the run for I-2.

That the constraint in the modified case I-1 ceases to evolve at some point is consistent with (3.7), which implies that the constraints will cease to evolve when the first term balances with the second term. From (3.7), one expects that this balance will be achieved for smaller constraint violation when the driving parameter is larger, which is consistent with the results shown in Figure 3.7.

Looking at two dimensional slices of the constraint data at various times, also

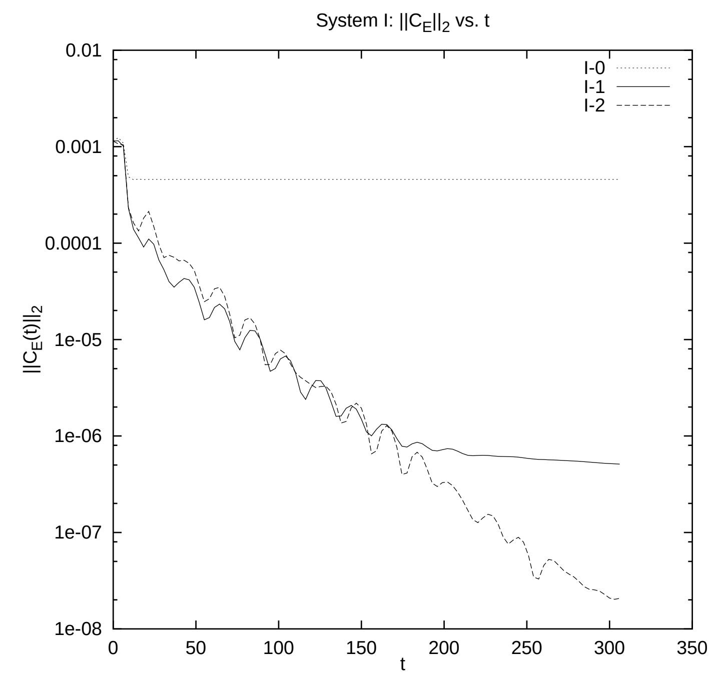


Figure 3.7: The l_2 norm of the primary constraint C_E versus time t for three test cases. Case I-0 has no correction terms ($K_E = 0$). See Table 3.1 for the other parameter values.

suggests that the source of the modulation in the decay demonstrated by I-1 and I-2 is fluctuations at the boundary, possibly caused by the simple boundary condition applied there on the constraints. Significantly, these fluctuations are unable to penetrate the interior of the computational domain, unlike many scenarios seen in numerical relativity where noise from the boundary noticeably propagates inward, eventually killing the simulation. Because I am only interested in the Maxwell equations as a test-bed for the method, and because the modified System I equations already perform orders of magnitude better than their unmodified counter part, I have not pursued this point further.

Figure 3.8 shows a plot of $\|C_E\|_2$ vs. t for the System II case. Here again, the control run (II-0) reproduces the results of KWB, this time showing exponential decay in the primary constraint. Even with such an ideal result in the unmodified case, the modified runs II-1 and II-2 show improvement. They represent runs with non-zero values (see Table 3.1) of the various forcing parameters, and in these cases

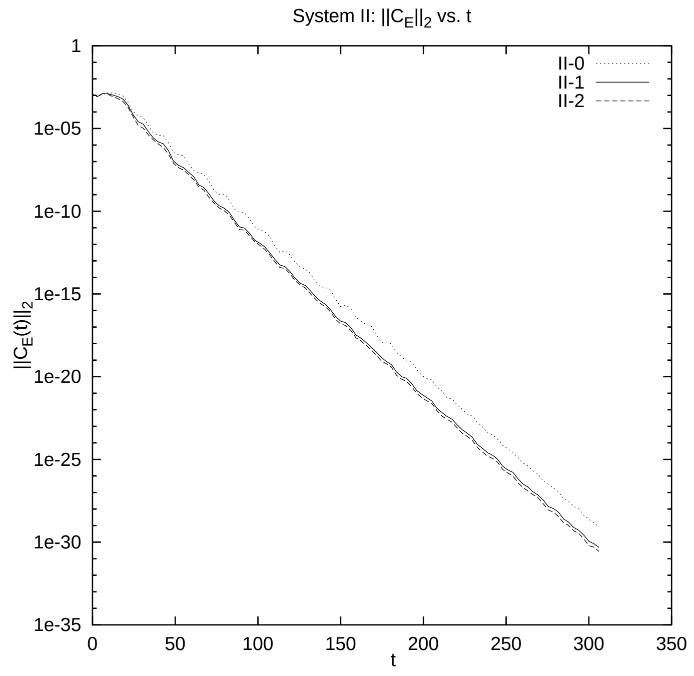


Figure 3.8: The l_2 norm of the primary constraint C_E versus time t for three test cases. Case II-0 reproduces KWB. See Table 3.1 for the definitions of other parameters.

the constraint decays exponentially, but with a smaller characteristic time.

Figure 3.9 is a plot of $\|C_\Gamma\|_2$ versus time t, and shows similar behavior to the primary constraint in all three cases. The secondary constraint shows exponential decay in the unmodified II-0 case, while showing a faster decay in the two modified runs, II-1 and II-2. The jump in the graph at $t = 0$ occurs because the secondary constraint is exactly satisfied in the initial data by construction.

It should be noted that one likely explanation for the extremely favorable performance of the unmodified System II is that, since the constraints satisfy a wave equation, constraint violations propagate off of the grid. This is supported by the data presented by KWB, showing that decay rate of the constraint decreases when the outer boundary is moved farther out [58]. The method presented here benefits from constraint violations propagating off of the grid as well, but, in addition, attempts to damp the constraint violation pointwise throughout the grid at all times.

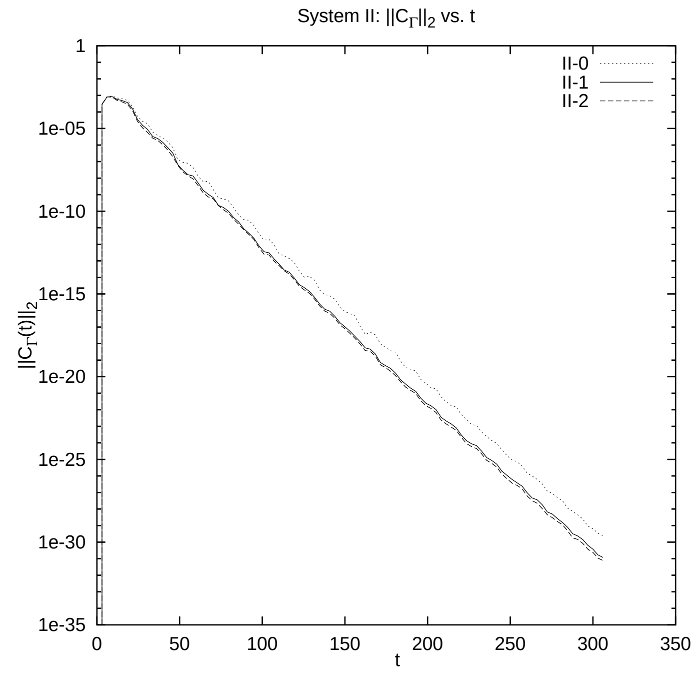


Figure 3.9: The l_2 norm of the secondary constraint C_Γ versus time t for three test cases. Case II-0 is the result of KWB. See Table 3.1 for other parameter values. At $t = 0$ the constraint is identically satisfied.

3.5 Application to the linearized Einstein Equations

The work detailed in the previous two sections provides the first proof-of-concept tests that driver terms can control constraint growth during the free evolution of a system of partial differential equations. The work of those sections, formally, generalizes simply to the more complicated Einstein equations. To preserve diffeomorphism symmetry in the equations and to properly balance the tensor density weights, the corrections should be computed as

$$\frac{\partial s_m}{\partial t} = S_m(t, \mathbf{x}) - \frac{\kappa_{mn}}{\sqrt{g}} \frac{\delta \overline{C^2}}{\delta s_n(t, \mathbf{x})} \tag{3.32}$$

instead of (3.6), letting $\overline{C^2} = \int C^2 \sqrt{g} d^N x$ now so that the natural volume element is used as the measure in the integration.

This method should now be tested in the specific context of an ADM decomposition of the Einstein equations for general relativity. Unlike the Maxwell equations, however, the Einstein equations are non-linear and have higher derivative terms in the constraints. In particular, on the issue of higher order derivatives, note that the correction terms generated by (3.32), for the formalisms that I am using here, will have fourth derivative terms after integrating by parts because the constraints themselves contain second derivatives.[5] Either the non-linearities or the higher order derivatives could cause serious complications when adding driver terms to the evolution equations. I began by studying the effect of of constraint driver terms by isolating potential issues coming from the higher order derivatives in the *linearized* Einstein equations, leaving the possible effects of non-linearities aside for a moment. Moreover, in this first round of tests, I study a simplified metric, which is assumed to be diagonal and depend only on time and a single spatial coordinate. I ignore the complications that fourth derivatives may introduce at the boundary by imposing periodic boundary conditions.

3.5.1 Linearized ADM

Considering a transverse linear plane wave propagating in a Minkowski background, the ADM evolution equations (2.12) take the form

$$\frac{\partial g_{ii}}{\partial t} = -2K_{ii} \tag{3.33a}$$

$$\frac{\partial K_{ii}}{\partial t} = -\frac{1}{2}g_{ii,xx} \tag{3.33b}$$

in the geodesic gauge ($\alpha = 1$, $\beta^i = 0$). In this section spatial indices run only over y and z, and the implicit summation convention is not used. There are two constraints (because two components of the momentum tensor are identically satisfied under the simplifying assumptions of the current problem)

$$H = -\frac{1}{2}(g_{yy,xx} + g_{zz,xx}) \tag{3.34a}$$

$$P = \frac{1}{2}(K_{yy,x} + K_{zz,x}) \tag{3.34b}$$

[5]In a fully first order system, the correction terms would only have second order derivatives, but this spoils the first order nature of the system!

which correspond to the linearized versions of (2.13) in the case of the transverse wave. I fix $g_{xx} = 1$ and $K_{xx} = 0$ because these tensor components do not appear in the constraints, and are therefore irrelevant to my current topic.

With these choices, it is easy to check that the ansatz

$$
\begin{aligned}
g_{yy} &= 1 + af(t,x) && (3.35a)\\
K_{yy} &= -\frac{a}{2}\frac{\partial f}{\partial t}(t,x) && (3.35b)\\
g_{zz} &= 1 - af(t,x) && (3.35c)\\
K_{zz} &= +\frac{a}{2}\frac{\partial f}{\partial t}(t,x) && (3.35d)
\end{aligned}
$$

identically satisfies the constraints (3.34) for any smooth function f. It is now sufficient to solve just one pair of evolution equations, say the g_{yy}-K_{yy} pair, to have a complete solution to the problem. Each pair, however, is a wave equation, so (3.35) is a solution to all of the linearized Einstein equations for any $f(t,x) = c_1 f_1(t+x) + c_2 f_2(t-x)$. For concreteness and for the numerical results below, I choose the particular solution

$$
f(t,x) = \sin((t+x)/\lambda) \qquad (3.36)
$$

where x runs over a periodic domain $[0, 2\pi\lambda)$.

Now I need to generate the correction terms for this formalism. Applying the prescription (3.32) to the constraints (3.34) combined into the equally weighted grand constraint functional[6] $\overline{C^2} = \int (H^2 + P^2) d^3x$,

$$
\begin{aligned}
\frac{\partial g_{ii}}{\partial t} &= -2K_{ii} - \kappa_g (g_{yy,xxxx} + g_{zz,xxxx}) && (3.37a)\\
\frac{\partial K_{ii}}{\partial t} &= -\frac{1}{2} g_{ii,xx} + \kappa_K (K_{yy,xx} + K_{zz,xx}) && (3.37b)
\end{aligned}
$$

replace (3.34) as the equations of motion for the system. I choose the parameters κ_g and κ_K to be positive constants. The goal, paralleling the argument with the Maxwell equations, is now to construct an explicit solution for the time evolution of the constraints (3.34) given the new evolution equations (3.37).

[6]In principle this should include a factor of $g^{1/2}$ as in (3.32), but I ignore that factor here because it does not contribute at first order.

As with Maxwell, this linear system of equations is simple enough for Maple to generate an explicit solution to the constraints' evolution equations, and, as with Maxwell, that solution indicates that arbitrary (unconstrained) initial data will approach the constraint hypersurface at late times. The full result is too long to display here, but has a form similar to (3.28) and (3.29). All terms in the solution are exponentially decaying with time.

I wrote a one dimensional numerical code to experiment with this system of equations. I specify initial data from the analytic solution (3.35) with the particular choice (3.36). All first and second spatial derivatives are computed using standard second order, centered finite difference stencils. The fourth derivatives are computed by applying the second derivative operator twice. That is

$$f^{(4)}(x_i) \approx \frac{D_2(f_{i-1}) - 2D_2(f_i) + D_2(f_{i+1})}{h^2} \tag{3.38}$$

where

$$D_2(f_j) = \frac{f_{j+1} - 2f_j + f_{j-1}}{h^2} \tag{3.39}$$

is the finite difference operator for second derivatives, and h is the grid spacing. This approximation is second order accurate. The time integration scheme is iterated Crank-Nicholson [87]. I find empirically that in order to have numerical stability with the constraint driver terms it is important to choose $dt = kh^2$ for some k rather than maintaining a more conventional Courant condition $dt = kh$. This was suggested, in the case of the Maxwell equations in Ref. [26]; in that case it was explicitly required because some of the modified equations are parabolic rather than hyperbolic. I choose $k = 1/4$. The wavelength λ is the natural length scale of the problem; I choose the amplitude $a = 10^{-6}\lambda$ in terms of that scale. With these choices, I verified that the code is second order convergent both with and without constraint driver terms.

Figure 3.10 shows the l_2 norms $\|E\|_2$ and $\|C\|_2$ for the linearized ADM system both with and without constraint driver terms. The error $\|E\|_2$ is defined to be

$$\|E\|_2 = \int \sum_{\text{fields } f} (f_{\text{numeric}} - f_{\text{analytic}})^2 dx \tag{3.40}$$

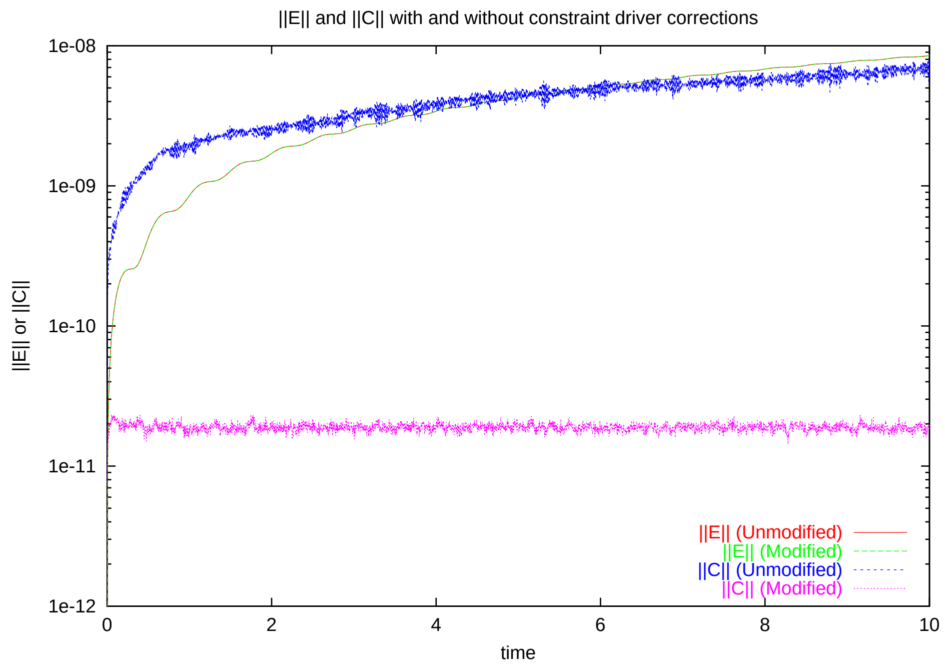


Figure 3.10: The figure shows the l_2 norm of the total solution error and the constraint violation $\|C\|_2$ for the linearized ADM system with and without constraint driver modifications. The two jagged lines show the constraint violations in the two cases. The jaggedness shows that the constraints are heavily influenced by round-off error. The two smooth lines showing $\|E\|_2$ for the two cases are completely overlapping.

and the constraint

$$\|C\|_2 = \int (H^2 + P^2) dx \tag{3.41}$$

weights the two physical constraints equally.

The results show several interesting features. Most prominently, they show that the constraint driver terms *do* reduce the constraint violation in the simulation. Because the norm of the error is the same regardless of whether or not the constraint driver terms are activated, I draw, in this case, two additional conclusions, (1) the constraint driver terms are not spoiling the solution,[7] and (2) there are significant

[7]One might worry, for example, that the constraints are solved, but that the resulting solution is not the solution to the constraints that corresponds to the initial data. This issue could be further addressed by considering the convergence of the simulation over several different choices of grid

constraint satisfying errors present in the simulation.

The fact that the constraint violations are reduced by the constraint driver terms without *increasing* the solution error is a necessary condition for the successful application of the algorithm, but what of the constraint satisfying errors? A few comments are useful on this point. First I should discuss what it means to have a constraint satisfying error. Recall that I noted above that (3.35) satisfies the constraint equations for *any* smooth function f. Not all choices of f, however, satisfy the evolution equations. If I make the particular choice

$$f(t,x) = \sin((vt + x)/\lambda) \tag{3.42}$$

where v is a constant, I will only have a solution to the evolution equations if $v = 1$ is the speed of light; compare (3.42) to (3.36). Choose any $v \neq 1$. Taking (3.35) with the choice (3.42) yields a one parameter family of solutions to the constraints, but the solutions at various "times" are not connected by the evolution equations. In other words, this "solution" satisfies the constraint equations at all times, but it never satisfies the evolution equations! Since numerical errors will generally violate both the constraint equations and the evolution equations, there should always be some "component" of constraint satisfying error in numerical results.

In the particular case of the plane wave solution simulated here, it seems that most of the error is constraint satisfying. This claim is supported by the fact that, as seen in Figure 3.10, the absolute error in the numerical answer is essentially unchanged when the constraint modifications are added, even though the constraint violation is reduced by two and a half orders of magnitude when the constraint driver terms are activated.

I should also say some words about the jaggedness of the lines showing $\|C\|_2$ in both the unmodified and the modified cases in Figure 3.10. This is understood analytically by noting that, in the initial data, the truncation error in the finite difference approximation to $g_{yy,xx}$ is exactly the same in magnitude and opposite in sign as the truncation error in the finite difference approximation to $g_{zz,xx}$ because

spacing.

of the symmetry between the equations. Specifically,

$$\left(\left.\frac{\partial^2 g_{yy}}{\partial x^2}\right|_{t=0}\right)_{\text{num}} = \left(\left.\frac{\partial^2 g_{yy}}{\partial x^2}\right|_{t=0} + \sum_{k=4}^{\infty} c_k h^{k-2} \left.\frac{\partial^k g_{yy}}{\partial x^k}\right|_{t=0}\right)_{\text{exact}} + e_y \tag{3.43}$$

$$\left(\left.\frac{\partial^2 g_{zz}}{\partial x^2}\right|_{t=0}\right)_{\text{num}} = \left(\left.\frac{\partial^2 g_{zz}}{\partial x^2}\right|_{t=0} + \sum_{k=4}^{\infty} c_k h^{k-2} \left.\frac{\partial^k g_{zz}}{\partial x^k}\right|_{t=0}\right)_{\text{exact}} + e_z \tag{3.44}$$

where the c_k are rational numbers, and the terms e_y and e_z represent the round-off errors in the calculation. Round-off errors are normally negligible. In this case, however,

$$\left(\frac{\partial^k g_{yy}}{\partial x^k}\right)_{\text{exact}} = -\left(\frac{\partial^k g_{zz}}{\partial x^k}\right)_{\text{exact}} \tag{3.45}$$

for all $k \geq 1$, which means that

$$(H)_{\text{num}} = (g_{yy,xx})_{\text{num}} + (g_{zz,xx})_{\text{num}} = e_y + e_z + e_H \tag{3.46}$$

where e_H is the round-off error accumulated in computing the constraint from the numerical derivatives. In other words, the constraint violation in the initial data is completely determined by round-off error. This phenomenon is a well-known feature of floating point arithmetic, and is called *catastrophic cancellation*. When two numbers of nearly equal size are subtracted, especially if those two numbers in reality differ only by round-off error, it frequently turns out that all of the significant digits cancel in the subtraction, leaving only round-off error behind. Because truncation error arising from the time integration causes the magnitude of $g_{yy,xx}$ and $g_{zz,xx}$ to evolve differently after the very earliest times, I do not see complete catastrophic cancellation in Figure 3.10. The jaggedness in the lines, however, indicates that round-off error continues to make a significant contribution to the constraint violation for all times shown. Evidently the symmetry between the x and the y equations forces this to persist even at later times.

In a moment, I will consider a different formalism, in which there are no such accidental cancellations in the constraints. There I will demonstrate, perhaps more convincingly because the constraint violations tend to be larger, that the constraint driver terms are behaving as designed for initially constrained data. Before doing that, however, consider the case of *unconstrained* data in the ADM formalism. In

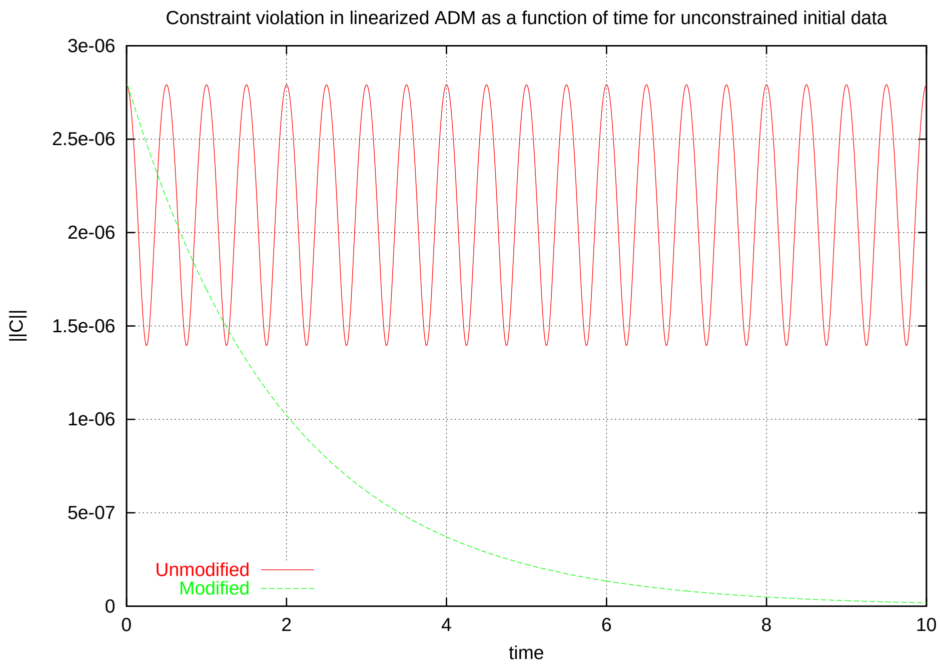


Figure 3.11: The figure shows the behavior of the total constraint violation for the ADM formalism for a particular choice of initial data that does *not* satisfy the constraints. Using the unmodified equations, the constraint violation oscillates, but over time remains essentially of the same amplitude. In the modified equations, however, the constraint violation rapidly damps away.

this case I kept the analytic solution (3.35) with the particular choice (3.36), and then perturbed the initial data for g_{yy} by giving it an amplitude of $a' = 1.1 \times 10^{-6}\lambda$. This amounts to a ten percent perturbation in this particular degree of freedom compared to the unperturbed fields which all have $a = 1.0 \times 10^{-6}\lambda$. Figure 3.11 shows the l_2 norm of the total constraint violation in both the unmodified system and the system with constraint driver terms. In the unmodified system, the constraint violation is oscillating, but has a fairly constant amplitude over time.[8] When the constraint driver terms are added, on the other hand, the constraint violation does damp away rapidly. Because the initial data in this case was not a solution to the Einstein equations, it is not possible to compute the error in the numerical solution

[8]This is consistent with the analysis in Ref. [58] and with (3.25a), which suggests that constraint violations in the "ADM" form of the Maxwell equations do not dissipate.

as compared to an analytic solution. At late times, however, the modified system finds a solution to the equations. Notice also that round-off error does not appear to be significant in this simulation (there are no jagged lines) because the symmetry between x and y has been explicitly broken by the choice of initial data.

3.5.2 A Linearized BSSN-Type System

Now I would like to study a case that does not have the accidental cancellation in the constraints that was present in the ADM system. To do this, I construct a new system of equations by defining a new variable

$$\Gamma = g_{yy,x} \tag{3.47}$$

and rewriting the ADM equations in terms of this new variable. Using (3.33) and (3.47), it is easy to derive new evolution equations

$$\begin{aligned}
\frac{\partial g_{yy}}{\partial t} &= -2K_{yy} && (3.48a)\\
\frac{\partial g_{zz}}{\partial t} &= -2K_{zz} && (3.48b)\\
\frac{\partial K_{yy}}{\partial t} &= -\frac{1}{2}\Gamma_{,x} && (3.48c)\\
\frac{\partial K_{zz}}{\partial t} &= -\frac{1}{2}g_{zz,xx} && (3.48d)\\
\frac{\partial \Gamma}{\partial t} &= -2K_{yy,x} && (3.48e)
\end{aligned}$$

for the fields in this new formalism. The constraints are (3.34) with the definition (3.47)

$$G = \Gamma - g_{yy,x} \tag{3.49}$$

rewritten so that it is zero valued.

Some remarks are needed. First, I call this system a "BSSN-type" system because the introduction of the new variable Γ shadows the introduction of the conformal connection Γ^i in (2.14e).[9] Second, it is clear that I have broken the symmetry

[9]The new variable that I am introducing here fails to capture some features of the BSSN system. In particular, the BSSN system introduces the momentum constraint into the evolution system

between the y direction and the z direction. While this seems undesirable in a broader context, keep in mind that I am introducing this system *specifically to break just that symmetry.* Without breaking the symmetry between y and z I would not be able to eliminate the accidental calculation that keeps the constraints so small in the linearized ADM system. Third, notice that, as always, there is no unique way to write the evolution and constraint equations. I could, for example, substitute $\Gamma_{,x}$ for $g_{yy,xx}$ in the Hamiltonian constraint (3.34a). I consistently, both in writing and in implementing, use the equations as described here.

Having made my specific choices on how to write the evolution equations and the constraints, I can now generate the correction terms. The modified equations of motion

$$\begin{aligned}
\frac{\partial g_{yy}}{\partial t} &= -2K_{yy} - \kappa_H(g_{yy,xxxx} + g_{zz,xxxx}) + \kappa_\Gamma(\Gamma_{,x} - g_{yy,xx}) && \text{(3.50a)}\\
\frac{\partial g_{zz}}{\partial t} &= -2K_{zz} - \kappa_H(g_{yy,xxxx} + g_{zz,xxxx}) && \text{(3.50b)}\\
\frac{\partial K_{yy}}{\partial t} &= -\frac{1}{2}\Gamma_{,x} + \kappa_P(K_{yy,x} + K_{zz,x}) && \text{(3.50c)}\\
\frac{\partial K_{zz}}{\partial t} &= -\frac{1}{2}g_{zz,xx} + \kappa_P(K_{yy,x} + K_{zz,x}) && \text{(3.50d)}\\
\frac{\partial \Gamma}{\partial t} &= -2K_{yy,x} - \kappa_\Gamma(\Gamma - g_{yy,x}) && \text{(3.50e)}
\end{aligned}$$

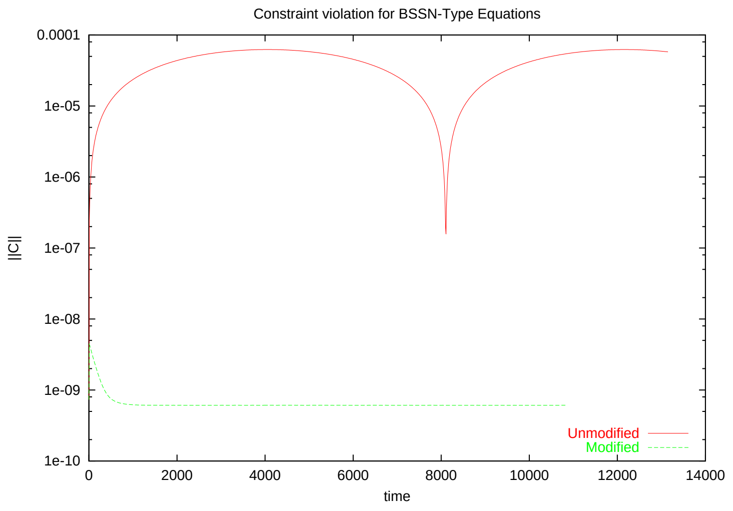
have positive, but otherwise free, constants κ_H, κ_P, and κ_Γ.

I also implemented this system in a 1D code. I continue to use the solution (3.35) with (3.36) to provide the initial data. Initial data for the new variable Γ is computed analytically from the definition (3.47). The numerical methods used in this code are identical to those in the 1D ADM code described in the previous subsection.

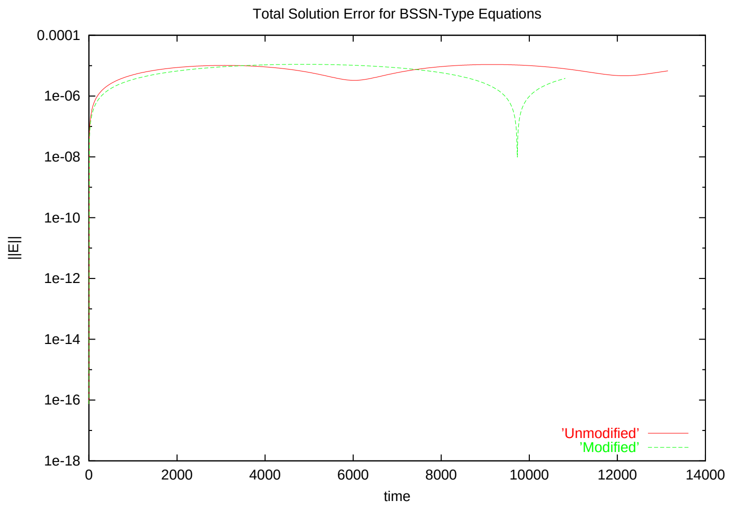
From Figure 3.12, which shows results from the 1D code, it is clear that the constraint violations in the new system are non-trivial and that round-off error is no longer playing a significant role. Panel (a) shows the l_2 norm of the total constraint

$$\|C\|_2 = \left[\int \left(H^2 + P^2 + G^2\right) dx\right]^{1/2} \tag{3.51}$$

through the evolution equation for Γ^i, which I am not doing with this system. While this seems to be a key feature of BSSN in real applications (see, e.g., [67]), it will prove irrelevant in this simplified test case.



(a)



(b)

Figure 3.12: The figure shows numerical results for the “BSSN-type” system with with a family of values for the damping parameters associated with the constraint driver terms. Panel (a) shows the l_2 norm of the total constraint violation, and Panel (b) shows the l_2 norm of the solution error, both as a function of time.

and Panel (b) shows the l_2 norm of the solution error analogous to (3.40).

Three of the lines in the figure show what appears to be the beginning of an oscillating pattern. This happens because the numerical solution suffers from a phase error. Because the domain is toroidal, however, the numeric and analytic solutions periodically experience an instant when they are in phase. When this happens, the absolute error dips down. For this reason, although the error appears to be an a minimum when the line dips down, it is in fact 2π radians out of phase at that time. This means, in particular, that although $\|E\|_2$ for the unmodified system registers lower than $\|E\|_2$ for the modified system at 6000 crossing times, the modified system is actually more accurate, and that this would be obvious if the domain were not periodic. One should look at the maximum amplitude and the number of peaks in the error function. Lower amplitude and fewer peaks means a better solution.

For the constraints, something similar is happening for the unmodified solution. The phase error in the various fields which contribute to the constraint is not the same. Because the domain is periodic, they nonetheless come momentarily into phase after a sufficient number of crossing times. Since the constraint is not sensitive to phase error (provided all fields have 2π-multiples of the same phase error), this causes a dip in the curve for $\|C\|_2$ in the unmodified case.

Of course, the most prominent feature of the figure is that the constraint violation in the modified case does decay rapidly away. It quickly settles to a value that is five orders of magnitude smaller than the peak value of the constraint violation in the unmodified case. And, from the figure, with the discussion above guiding the analysis, it is clear that it reduces the constraint violation while simultaneously reducing the total solution error. This is the best possible result for my method.

3.6 Summary and Future Work

In this chapter I have introduced and studied a novel technique modifying equations of motion such the dynamics of the system favor the constraint hypersurface. The prescription is well defined for any system of partial differential equations that can be split into evolution and constraint equations. Unlike other authors who have studied constraint subtraction methods, the method provided here is generically non-linear, although it has yet to be tested on a non-linear problem.

I have applied this method to the Maxwell equations, written in two different formalisms, as well as to the linearized Einstein equations. In both cases numerical evidence supports the theoretical prediction that the constraint driver terms will damp constraint violating modes generated by numerical errors in the simulations. In the case of the Einstein equations the prescription presented here produces correction terms that contain fourth derivatives of the numerically evolved fields. Although a priori such terms might cause concern about the stability of the resulting system, the available numerical evidence indicates that such terms do not create stability problems.

It is also important to appreciate that the constraint driver terms damp the constraint violations without increasing the solution error, and in many cases actually improve the overall quality of the simulation. This is significant. One might worry that the driver terms would find the wrong solution to the constraint equations. All available evidence on this point suggests that the method that I am proposing here does not suffer from that potential problem.

When this method is applied to the full (non-linear) Einstein equations, the correction terms will also be non-linear. Moreover, the correction terms will also not be, in general, quasi-linear, and there is relatively little known mathematically about such systems. In lieu of detailed mathematical understanding of how constraint driver terms will effect the numerical properties of the Einstein system, and encouraged by the results in linearized systems, I would like to continue the study started here by a combination of analytical work and numerical experiments. This should include work with the constraint driver terms in the context of the non-linear Einstein equations in one dimension (spherical symmetry) first, and, if the results are positive, should continue to the three dimensional equations. In the event that the results are unfavorable for the method as written, it may also be interesting to consider the linearized corrections applied to the non-linear equations. While less attractive in the sense that such a prescription would require re-tuning the constraint driver terms for each spacetime simulated (since one would, presumably, want the corrections linearized around a background adapted to the problem at hand), it might avoid any complications that arise from spoiling quasi-linearity.

Chapter 4

Gravitational Wave Extraction

In order for numerical relativity to make contact with gravitational wave experiments, numerical relativists must be able to compute gravitational waves within the framework of their numerical simulations. This has, historically, proven to be a difficult task in practice partially because simulations have only recently become stable enough for wave extraction to be meaningful and partially because the computational requirements of a full, three-dimensional simulation have made it difficult or impossible to extend the simulation domain into the wave zone while simultaneously resolving the sources.

This chapter briefly covers the formalism required for defining waves, which means specifically the Newman-Penrose formalism applied in a 3+1 spacetime, as well as the techniques used for decomposing the Weyl scalars into spherical harmonic components. After this review, I present, for the first time, a detailed error analysis for a new algorithm, due to Misner, for computing spherical harmonic components on a cubic grid. Finally, I demonstrate that this technology works in three dimensional simulations of linearized gravity both with and without fixed mesh refinement. Because FMR can be used to extend the spatial domains of simulations beyond what is possible in unigrid domains, this demonstration is a key step toward extracting waveforms from more realistic sources.

4.1 Newman-Penrose Formalism

This section does not attempt to treat the Newman-Penrose formalism in its entirety. The purpose is to highlight the part of the formalism that is essential for computing and extracting gravitational waves from numerical simulations. Readers interested in more detail should consider reading Chapter 1 of Chandrasekhar [28] and the references therein. After outlining the basics of the formalism, I provide, both as historical context and as an introduction to later sections, the rationale for adopting this particular formalism in the discussion of gravitational waves.

4.1.1 Tetrads and Weyl Scalars

The Newman-Penrose approach requires choosing four null vectors, the *tetrad*, with respect to which all other quantities will be referred. Two of the vectors, l^a and n^a are real valued, while the remaining two m^a and $\bar{m}^a$ are complex and conjugate to each other.[1] The vectors l^a and n^a are chosen to point along "outgoing" and "ingoing" directions respectively (defined more rigorously in what follows), and the vectors m^a and $\bar{m}^a$ are chosen along angular directions. These are always chosen to satisfy certain orthogonality conditions

$$l^a m_a = n^a m_a = 0 \tag{4.1}$$

and, in addition, may be normalized

$$\begin{aligned} l^a n_a &= -1 && (4.2\text{a}) \\ m^a \bar{m}_a &= +1 && (4.2\text{b}) \end{aligned}$$

to remove some arbitrary freedom from their definition.[2] Note that the angular vectors m and $\bar{m}$ are still somewhat arbitrary; they may be rigidly rotated without

[1]When referring to Newman-Penrose quantities, the indices are understood to be *four* dimensional unless otherwise noted.

[2]I follow Chandrasekhar and earlier work of Penrose in imposing normalization conditions (4.2) because there is no advantage in this work to allowing extra freedom. In other contexts, conditions (4.2) are not applied. Also notice that, because Chandrasekhar writes his metric with the opposite signature, he has opposite signs on his version of (4.2) in Ref. [28].

violating any of the conditions laid out thus far. The physical implications of this are discussed below.

With a tetrad specified, it becomes possible to recast various quantities familiar in the usual coordinate basis formalism into Newman-Penrose quantities. In particular, one can always contract indices of tensors with tetrad vectors to form scalars. The relevant example for wave extraction comes from the Weyl tensor C_{abcd}, which is recast into five complex scalars

$$\begin{aligned}
\Psi_0 &= -C_{pqrs} l^p m^q l^r m^s &\quad (4.3a)\\
\Psi_1 &= -C_{pqrs} l^p n^q l^r m^s &\quad (4.3b)\\
\Psi_2 &= -C_{pqrs} l^p m^q \bar{m}^r n^s &\quad (4.3c)\\
\Psi_3 &= -C_{pqrs} l^p n^q \bar{m}^r n^s &\quad (4.3d)\\
\Psi_4 &= -C_{pqrs} n^p \bar{m}^q n^r \bar{m}^s. &\quad (4.3e)
\end{aligned}$$

These *Weyl scalars* are coordinate invariant, but tetrad dependent.

In the case of the Weyl scalars, recasting to the Newman-Penrose formalism is especially useful. It can be shown that the various Weyl scalars, when the tetrad is oriented in a conventional way, fall off with different powers of radial coordinate. It is Ψ_4 that falls off as $1/r$ and that represents outgoing radiation, and it is Ψ_4 therefore on which we will concentrate in the rest of this chapter.

For any practical computation, it becomes important to choose a reasonable tetrad. For the Kerr spacetime, written in Boyer-Lindquist coordinates

$$ds^2 = -\left(1 - \frac{2Mr}{\Sigma}\right) dt^2 + \frac{\Sigma}{\Delta} dr^2 + \Sigma d\theta^2 + \frac{\Omega \sin^2\theta}{\Sigma} d\phi^2 - \frac{4aMr\sin^2\theta}{\Sigma} dt d\phi \quad (4.4)$$

with $\Delta = r^2 - 2Mr + a^2$, $\Sigma = r^2 + a^2\cos^2\theta$, and $\Omega = (r^2 + a^2)\Sigma + 2Mra^2\sin^2\theta$ in terms of black hole mass M and spin per unit mass a, a conventional choice for the tetrad has an easy expression. This tetrad, sometimes called the Kinnersley tetrad [57], is motivated physically by choosing l^a and n^a to point, respectively, along the outgoing and ingoing null geodesics, and choosing the m^a-$\bar{m}^a$ pair orthogonal. In

Boyer-Lindquist coordinates, these vectors are

$$l = \frac{1}{\Delta}(r^2 + a^2, \Delta, 0, a) \tag{4.5a}$$

$$n = \frac{1}{2\Sigma}(r^2 + a^2, -\Delta, 0, a) \tag{4.5b}$$

$$m = \frac{1}{\sqrt{2}(r + ia\cos\theta)}(ia\sin\theta, 0, 1, i\csc\theta) \tag{4.5c}$$

with the normalization according to (4.2). Notice, in particular, that while l^a is affinely parameterized, n^a is not [28, 10].[3]

For Kerr spacetimes, the coordinate expressions on the right hand sides of (4.5) are only valid in Boyer-Lindquist coordinates. More generally, they are not, strictly speaking, meaningful at all for spacetimes other than Kerr. Nonetheless, the Weyl scalars computed in the numerical results presented below are computed with respect to a tetrad constructed from the coordinate expressions in (4.5). For weakly distorted Kerr solutions, this should not introduce significant error. If, at some later time, this proves insufficient, it would be possible to use a Gramm-Schmitt procedure to construct a tetrad more adapted to the numerical coordinates as has been successfully done in work that patches non-linear evolutions to a perturbation code [10].

The Weyl scalars, as written in (4.3), depend on the Weyl tensor of the full *four* dimensional metric. In order to compute them on a single time slice in a numerical simulation, it is most convenient to rewrite them in terms of data available on a three dimensional surface.[4] Noting that, for vacuum spacetimes, the Riemann and Weyl

[3]The first component of (4.5c) differs from equation 5.4c in Ref. [10], This appears to be a typographical error in Ref. [10].

[4]One could, alternatively, compute the time derivatives necessary by finite differencing between data on different time slices, but this requires saving more data in simulations which already push the physical limits of available computers.

tensors are equal, the following expressions[5]

$$
\begin{aligned}
{}^{(4)}R_{ijkl} &= R_{ijkl} + 2K_{i[k}K_{l]j} && (4.6a)\\
{}^{(4)}R_{0jkl} &= -2\left(K_{j[k,l]} + \Gamma^{p}_{j[k}K_{l]p}\right) && (4.6b)\\
{}^{(4)}R_{0j0l} &= R_{jl} - K_{jp}K^{p}_{\ l} + KK_{jl} && (4.6c)
\end{aligned}
$$

in which everything on the right hand sides is a three dimensional quantity, are useful [10].

The Newman-Penrose formalism turns out to be an ideal framework for perturbation studies in general relativity. Price was the first to show that all non-trivial perturbations of a Schwarzschild black hole are described by Ψ_0 and Ψ_4, and, further, that the equations for these quantities, in the Schwarzschild background, separate and decouple [72]. Teukolsky showed that the same is true for Kerr spacetimes of arbitrary spin [85]. Knowing now that Ψ_4, for example, represents outgoing radiation, it is also possible to clarify the physical significance of it being complex valued. The real and imaginary parts correspond to the two possible polarizations of gravitational radiation. Moreover, this casts light on the fact, noted above, that the tetrad vectors m and $\bar{m}$ are oriented in an arbitrary manner. Mathematically such a rotation mixes the real and imaginary parts of Ψ_4, but this mixing corresponds exactly to reorienting a theoretical gravitational wave detector by half of the same angle.

4.1.2 Spin-Weighted Spherical Harmonics

Although the Weyl scalars, for example, are coordinate scalars, they retain a tensor-like quality. To be more precise, consider the effects on Ψ_0 and Ψ_4 of a rotation

$$m' = e^{i\chi}m \tag{4.7}$$

of the space-like vectors $\mathrm{Re}\,\{m\}$ and $\mathrm{Im}\,\{m\}$ through an angle χ. Substituting (4.7) into (4.3a), yields

$$\Psi_0' = e^{2i\chi}\Psi_0 \tag{4.8}$$

[5]Note that there is a factor of two difference between (4.6b) and Ref. [10]. This appears to be an error in Ref. [10].

whereas substituting (4.7) into (4.3e) yields

$$\Psi_4' = e^{-2i\chi}\Psi_4 \tag{4.9}$$

instead. Evidently Ψ_0 and Ψ_4 transform differently under such a rotation even though they are both nominally coordinate scalars. (Although they are *coordinate* scalars, they remain tetrad dependent, which is the key fact appearing here.) In general, a field η that transforms under a rotation (4.7) as

$$\eta' = e^{is\chi}\eta \tag{4.10}$$

will be said to have *spin-weight s*. With this definition, (4.8) and (4.9) clearly indicate that Ψ_0 is of spin-weight $+2$, while Ψ_4 is of spin-weight -2.

Because fields of different spin-weights, by definition, transform differently under rotations, it is not entirely surprising that they require different treatments when it comes to extracting spherical harmonic components. It is *spin-weighted* spherical harmonics, rather than the usual spherical harmonics, that are appropriate for describing spin-weighted fields. The spin-weighted harmonics are constructed to transform as scalars of a given spin-weight. Although expanding any function on the sphere in terms of spherical harmonics of the "wrong" weight is a mathematically well defined operation, using the "right" weight is more convenient. This is very similar to describing two orbiting bodies in a co-rotating coordinate frame rather than an inertial frame. Because the goal of the coming sections will be to extract spherical harmonics modes from Ψ_4 in numerical simulations, it is important to fully define spin-weighted spherical harmonics, as well as listing a few key properties. The discussion of this subsection follows Refs. [69] and [45], and adopts the convention that the angular vectors $\mathrm{Re}\{m\}$ and $\mathrm{Im}\{m\}$ point along the coordinate lines of θ and ϕ.

Consider a field η of spin-weight s defined on a sphere with angular coordinates $(\theta,\,\phi)$. Define the operator

$$\eth\eta = -(\sin\theta)^s\left(\frac{\partial}{\partial\theta} + i\csc\theta\frac{\partial}{\partial\phi}\right)(\sin\theta)^{-s}\eta \tag{4.11}$$

for functions on the sphere.[6] From this definition, it is easy to check that the transformation

$$(\eth\eta)' = e^{i(s+1)\chi}(\eth\eta) \tag{4.12}$$

holds under the rotation (4.7). A factor of $e^{is\chi}$ comes from the assumed transformation property of η, and an additional factor of $e^{i\chi}$ comes from the fact that the operator in parenthesis is m contracted with the natural derivative operator on the sphere. The $\eth$ operator is, apparently, a raising operator on the spin-weight "quantum number." The operator

$$\bar{\eth}\eta = -(\sin\theta)^{-s}\left(\frac{\partial}{\partial\theta} - i\csc\theta\frac{\partial}{\partial\phi}\right)(\sin\theta)^{s}\eta \tag{4.13}$$

is the corresponding lowering operator.

With these ladder operators, spin-weighted spherical harmonics ${}_sY_{lm}$ of any spin-weight s are defined in terms of the scalar spherical harmonics ${}_0Y_{lm} \equiv Y_{lm}$. The explicit definition for integral spin, which is the only case of interest here,

$${}_sY_{lm} = \begin{cases} \left[\frac{(l-s)!}{(l+s)!}\right]^{1/2} \eth^{+s}Y_{lm} & 0 \le s \le l \\ (-1)^s\left[\frac{(l+s)!}{(l-s)!}\right]^{1/2} \bar{\eth}^{-s}Y_{lm} & -l \le s \le 0 \end{cases} \tag{4.14}$$

includes prefactors that normalize the spin-weighted harmonics. The spin-weighted harmonics are not defined when $l < |s|$. It can be proved by induction on s that the spin-weighted spherical harmonics of any fixed s form a complete, orthonormal set on the sphere. That is

$$\oint {}_s\bar{Y}_{lm}(\theta,\phi)\, {}_sY_{l'm'}(\theta,\phi)d^2\Omega = \delta_{ll'}\delta_{mm'} \tag{4.15}$$

for any s. The specific case of interest in the applications that follow is in ${}_{-2}Y_{20}$. Given that

$$Y_{00}(\theta,\phi) = \sqrt{\frac{1}{4\pi}} \tag{4.16}$$

and

$$Y_{20}(\theta,\phi) = \sqrt{\frac{5}{16\pi}}(3\cos^2\theta - 1) \tag{4.17}$$

[6]The symbol $\eth$ is called eth and is conventional for this operator.

it is easy[7] to apply (4.14) to find

$$ {}_{-2}Y_{20}(\theta,\phi) = \sqrt{\frac{15}{32\pi}}\sin^2\theta = \sqrt{\frac{5}{6}}\left(Y_{00}(\theta,\phi) - \frac{1}{\sqrt{5}}Y_{20}(\theta,\phi)\right) \tag{4.18} $$

with the second equality, specifying how to convert between spin-weight -2 and scalar (spin-weight 0) spherical harmonics. As a further check on my work, I will compute

$$ {}_{-2}Y_{30}(\theta,\phi) = \frac{15}{4}\sqrt{\frac{7}{30\pi}}\left(5\cos^3\theta - 3\cos\theta\right) = \frac{1}{10}\left(-\sqrt{70}Y_{10}(\theta,\phi) + \sqrt{30}Y_{30}(\theta,\phi)\right) \tag{4.19} $$

in a few applications as well. Because my code is constructed to compute scalar spherical harmonics, I make use of these conversion relations between spin-weighted and scalar harmonics in Section 4.3.2.

4.2 Spherical Harmonic Decomposition

In order to analyze potential waveforms, it is frequently useful to look at the spherical harmonic decomposition of Ψ_4. Its spherical harmonic components $\Psi_{4,lm} = \Psi_{4,lm}(t,r)$ are defined by

$$ \Psi_{4,lm}(t,r) = \oint {}_{-2}\bar{Y}_{lm}(\theta,\phi)\Psi_4(t,r,\theta,\phi)d^2\Omega \tag{4.20} $$

in terms of a spin-weighted spherical harmonic. Because the spin-weighed harmonics can be re-expressed in terms of regular spherical harmonics (cf. (4.18)), I will opt to compute

$$ {}_0\Psi_{4,lm}(t,r) = \oint \bar{Y}_{lm}(\theta,\phi)\Psi_4(t,r,\theta,\phi)d^2\Omega \tag{4.21} $$

directly from the evolved data, and compute the spin-weighted components from the unweighted components.[8]

[7]It is easy, but one should keep in mind that when computing $\bar{\eth}^2 Y_{20}$ the first time that $\bar{\eth}$ acts, it acts on a spin-weight 0 quantity, whereas the second time that $\bar{\eth}$ acts, it acts on a spin-weight -1 quantity.

[8]I make this choice because, having implemented the specific case of scalar spherical harmonics as a proof-of-concept test, I wish to exploit that code while developing code to compute spin-weighted harmonics directly.

Whichever components one chooses to compute directly, from a numerical point of view, given that data is typically represented on a Cartesian, cubic grid, the integration over a *sphere* in (4.20) or (4.21) is extremely inconvenient. In order to perform such integrals, previous authors have resorted to interpolating from the cubic grid to a spherical grid, and performing a discrete integral on the interpolated data. While there is nothing wrong with this approach in principle, developing a robust framework for approaching the problem this way takes a considerable amount of work, whereas the algorithm developed below has relatively modest costs in code-development time. It also has the run-time advantage, in simulations where the extraction radii remain fixed, that the overhead associated with interpolation is paid only once when the numerical grid is created rather than each time a spherical harmonic component is computed.

In the rest of this section I describe, theoretically, one particular method for computing spherical harmonic components on a cubic grid. The details of my particular implementation of this algorithm are included in Appendix B.

4.2.1 Methodology

The idea used here follows that of Misner [65]. Although the development here is written in terms of the usual spherical harmonics (as it is in Ref. [65]), the mathematical foundations of the method generalize any to family of spin-weighted harmonics. The only specific property of the spherical harmonics used in the development of the algorithm is that they form a complete orthonormal set on the sphere, but in light of (4.15), spin-weighted spherical harmonics would work as well.

In what follows, I outline in some detail the steps of the algorithm, although I will not provide a full derivation. (See Ref. [65] for the derivation). A basic outline of the algorithm is essential, since I will build on Misner's results with more extensive analysis of the numerical errors associated with this decomposition method.

In order to begin, two definitions are need. First define a radial function

$$R_n(r; R, \Delta) = r^{-1}\sqrt{\frac{2n+1}{2\Delta}}P_n\left(\frac{r-R}{\Delta}\right) \tag{4.22}$$

in terms of the usual Legendre polynomials P_n. Here R and Δ are parameters that will be associated with the radius at which the spherical harmonic decomposition is desired and half of the thickness of a shell centered on that radius. From this, define

$$Y_{nlm}(r,\theta,\phi) = R_n(r)Y_{lm}(\theta,\phi) \tag{4.23}$$

which form a complete, orthonormal set with respect to the inner product

$$\langle f|g\rangle = \int_S \bar{f}(x)g(x)d^3x \tag{4.24}$$

on the shell $S = \{(r,\theta,\phi) \mid r \in [R-\Delta, R+\Delta]\}$. Note also that, because the functions R_n form a complete set on the shell,

$$\Psi_{4,lm}(t,R) = \int \left[\sum_{n=0}^{\infty} R_n(R;R,\Delta)R_n(r;R,\Delta)\right] \bar{Y}_{lm}(\theta,\phi)\Psi_4(r,\theta,\phi)d^3x \tag{4.25}$$

and that the term in brackets

$$\sum_{n=0}^{\infty} R_n(R;R,\Delta)R_n(r;R,\Delta) = r^{-2}\delta(R-r) \tag{4.26}$$

is a delta function. (Compare (4.25) to (4.21).)

On a finite grid Γ, the inner product (4.24) will have the form

$$\langle f|g\rangle = \sum_{x\in\Gamma} \bar{f}(x)g(x)w_x \tag{4.27}$$

where each point has some weight w_x. This weight was given the form

$$w_x = \begin{cases} 0 & |r-R| > \Delta + h/2 \\ h^3 & |r-R| < \Delta - h/2 \\ (\Delta + h/2 - |R-r|)h^2 & \text{otherwise} \end{cases} \tag{4.28}$$

by Misner, where h is the grid spacing.[9] Only cases with $\Delta > h/2$ are considered. This means, roughly, that points entirely within the shell S are weighted by their finite volume on the numerical grid, points entirely outside of the shell S have zero

[9]Misner only considered the unigrid case in Ref. [65]. In Section 4.2.5 and Appendix B I indicate how I implemented the method for FMR, and in Section 4.3.2 I show numerical results from an application in FMR.

weight, and points near the boundary are weighted according to the fraction of their volume inside S.[10]

With the numerical inner product (4.27), and letting capital Roman letters $A = (nlm)$ represent index groups, the Y_A are no longer orthonormal. Their inner product

$$\langle Y_A | Y_B \rangle = G_{AB} = \bar{G}_{BA} \tag{4.29}$$

forms a metric for functions on the shell. Although a priori this matrix appears to be complex valued, I will show in Section 4.2.4 that is actually real-symmetric and sparse. For now it suffices to follow Misner in denoting it as generically Hermitian. The inverse to this metric G^{AB} can be used to raise indices on functions defined on the sphere.

Making use of this new metric, and with some further analysis, the approximation for the spherical harmonic coefficients

$$\Psi_{4,lm}(t, R) = \sum_{x \in \Gamma} \bar{R}_{lm}(x; R) w_x \Psi_4(t, x) \tag{4.30}$$

follows with

$$R_{lm}(r; R) = \sum_{n=0}^{N} \bar{R}_n(R) Y^{nlm}(r, \theta, \phi) \tag{4.31}$$

in terms of $Y^A = G^{BA} Y_B$, not Y_A.

4.2.2 Error Analysis

In Ref. [65], Misner provides an algorithm for computing spherical harmonic components on a cubic lattice, but he does not provide any analysis of the errors involved. In a practical application of the method, such error analysis, in particular an analysis of the convergence order in grid spacing, is very valuable. For this reason, I wish to examine the issue more closely here.

[10] Actually the boundary points are weighted by the fraction of their volume that would be inside S if the point were on a coordinate axis. See Ref. [65] for more details on the definition of the weights.

To begin the analysis, note that there are three parameters in the algorithm, the grid spacing h, the width of the shell Δ, and the number of terms in the sum over basis polynomials N. (Compare (4.25) to (4.30) and (4.31).) In the limit that $h \to 0$ and $N \to \infty$, the numerical algorithm goes to the continuum theory. Note that, when h and N go to their limits, *any* value for Δ is allowed, since (4.25) is an exact, continuum expression. For a finite value of N, however, the quality of the approximation in the radial direction is a function of both N and Δ. Moreover, tying the value of Δ to the grid spacing h, though not necessary from fundamental considerations, does in fact have advantages in terms of convergence behavior that will become clear below.

The convergence of the method as grid spacing h goes to zero depends primarily on the order of accuracy in the volume integral when using the weights defined by (4.28). For a finite volume, this weighing scheme provides an approximation that scales as $\mathcal{O}(h)$, but, for a region that is also scaling with h, the resulting integral scales as $\mathcal{O}(h^2)$. This provides a strong motivation for choosing $\Delta \propto h$ since the simulations described in this dissertation are all second order accurate. Having prescribed the method in just this way, Misner allows for a second order convergent method, although the effects of N and Δ need to be considered before making a firm conclusion.

Having just decided that I should choose $\Delta \propto h$, I now consider the convergence behavior of the algorithm as $\Delta \to 0$ for fixed values of N. Define

$$d(x; N, R, \Delta) = \sum_{n=0}^{N} \frac{2n+1}{2\Delta} P_n\left(\frac{x-R}{\Delta}\right) P_n(0) \tag{4.32}$$

which is closely related to the delta function (4.26) in the limit $N \to \infty$. For finite N, this should approximate the delta function to some order of accuracy. Figure 4.1 gives a feel for how quickly the function d goes to a delta function for increasing N. It seems clear that the first few terms make the biggest individual contributions to the sum, suggesting that later terms in the series may not be so important. To quantify this, consider the approximation

$$f(0) \approx I_N[f] \equiv \int_{-\Delta}^{\Delta} d(x; N, 0, \Delta) f(x) dx \tag{4.33}$$

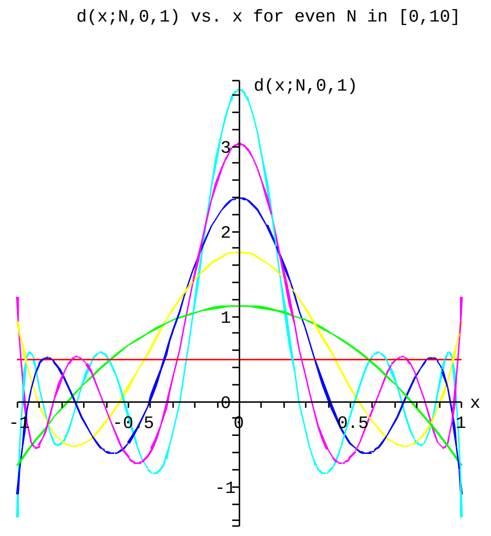


Figure 4.1: A plot of $d(r; N, 0, 1)$ for various even values of N in [0,10]. The central peak grows monotonically with N. Although in the limit of infinite N, d goes to a delta-function, it is clear that the function profile changes dramatically only for the first few values of N.

for a suitably smooth test function f. In the limit that $N \to \infty$, this is exact. For any finite N, this expression can be analyzed by Taylor expanding the test function around the point $x = 0$. This gives, after rearranging terms and noting that all of the odd terms vanish by symmetry,

$$I_N[f] = \sum_{k=0}^{\infty} \frac{c_{N,2k}}{(2k)!} f^{(2k)}(0) \tag{4.34}$$

where

$$c_{N,k}(\Delta) = \int_{-\Delta}^{\Delta} x^k d(x; N, 0, \Delta) dx \tag{4.35}$$

are the coefficients of the expansion. In order to have a high order method, I need $c_{N,0} = 1$, and the coefficients corresponding to the next few values of k to vanish. Table 4.1 shows the first few coefficients $c_{N,k}(1) = \Delta^{-k} c_{N,k}(\Delta)$ for even N in [0,10]. It is clear that each time N is increased, one more of the sub-leading coefficients vanishes. For a second order accurate method, it is sufficient to take $N = 0$ (provided that $\Delta \propto h$).

$N\backslash k$	0	2	4	6	8	10	12
0	1	1/3	1/5	1/7	1/9	1/11	1/13
2	1	0	-3/35	-2/21	-1/11	-12/143	-1/13
4	1	0	0	5/231	5/143	6/143	10/221
6	1	0	0	0	-7/1287	-23/2431	-70/4199
8	1	0	0	0	0	63/46189	15/4199
10	1	0	0	0	0	0	-33/96577

Table 4.1: The first few non-trivial values of $c_{N,k}(1)$, which are the coefficients of the Taylor expansion of a function integrated against $d(x; N, 0, 1)$. (In general, $c_{N,k}(\Delta) = c_{N,k}(1)\Delta^k$.) The values of k run across and the values of N run down. The fact that the first coefficient is always 1, and that, by increasing N, more of the sub-leading coefficients are 0 indicates that increasing N increases the order of convergence of the Misner algorithm (provided that $\Delta \propto h$).

4.2.3 Choosing the parameters

In practice, the grid spacing parameter h is usually chosen to resolve the sources without exceeding the physical limits of the computer. I would not expect, in general, that the grid spacing would be chosen based on the needs of this algorithm. For that reason, let me assume now that h is chosen, and discuss how to choose the remaining parameters N and Δ. In this subsection I will discuss some of the theoretical issues that should be considered when choosing the parameters, leading to a rule of thumb that is valid based on this analysis and my experience with the algorithm.

The error analysis of Section 4.2.2 implies that for fixed Δ, increasing the value of N decreases the error term. It also implies that for fixed N, increasing the value of Δ increases the error term. This suggests taking Δ as small as possible, and N as large as possible to make the error term as small as possible. This must be balanced, however, against practical limitations. Certainly the shell thickness Δ needs to be large enough so that there are some grid points within the shell, otherwise the whole procedure is undefined. For fixed N, a stronger restriction requires that the Legendre polynomial P_N can be resolved over the shell. Without this condition, there would seem to be no benefit to taking higher values of N. Getting higher accuracy in practice requires finding a proper balance between choosing Δ small and N large.

In making this balance, however, one must keep in mind that the error in the method is partially determined by the weighting scheme (4.28), which is only second order accurate in the grid spacing. I am, in addition, going to choose $\Delta \propto h$ for reasons described above. This already suggests that taking N larger than two is pointless, since choosing $N = 2$ already makes the piece of the error that is proportional to Δ scale like $\mathcal{O}(\Delta^4)$ (cf. Table 4.1), meaning that it will be an error term of sub-leading order in grid spacing. But once this term is of sub-leading order, it is much less important how large I choose Δ, provided that I still choose it proportional to the grid spacing. I therefore adopt the following

> **Rule of Thumb**: Choose N just large enough to ensure that the error term proportional to Δ is an error term of sub-leading order in grid spacing. Choose Δ just large enough to safely resolve P_N on the shell.

With this rule of thumb, and the second order accurate weighting scheme (4.28), I found the choices $N = 2$ and $\Delta = 3h/4$ completely satisfactory. Note that this corresponds to Misner's choice of Δ in Ref. [65]. With $N = 2$ I found that larger values of Δ are also acceptable. Numerical results justifying these estimates will come in Section 4.3.2.

4.2.4 Symmetry issues

There are two points of interest related to this method of spherical harmonic decomposition and symmetries. The first was mentioned briefly in Section 4.2.1, namely that symmetry causes the metric G_{AB} to be real and sparse. The second deals with implementing the method for grids in which explicit symmetries are enforced on grid functions in order to reduce the computational load of the simulation. In these cases, in which data is not evolved over a whole extraction sphere, additional analysis is required to demonstrate that the method is well defined and to understand how to most efficiently implement it. The primary result on this second topic is that the adjoint harmonics Y^A of (4.31) have the same symmetries under reflection as the original spherical harmonics Y_A.

Planes	θ	ϕ	**Sign**	**Conjugate**
None	θ	ϕ	$+1$	no
x	θ	$\pi - \phi$	$(-1)^m$	yes
y	θ	$-\phi$	$+1$	yes
z	$\pi - \theta$	ϕ	$(-1)^l$	no
xy	θ	$\pi + \phi$	$(-1)^m$	no
xz	$\pi - \theta$	$\pi - \phi$	$(-1)^{l+m}$	yes
yz	$\pi - \theta$	$-\phi$	$(-1)^l$	yes
xyz	$\pi - \theta$	$\pi + \phi$	$(-1)^{l+m}$	no

Table 4.2: The table shows how the arguments of spherical harmonics transform under reflections through various Cartesian planes. The first column indicates which coordinates have their signs inverted, while the second and third columns give the new angular arguments to the spherical harmonic Y_{lm}. Alternatively, the fourth and fifth column show, respectively, the overall sign in front of and whether or not to complex conjugate the given spherical harmonic with the original angular arguments. The second row, for example, says that $Y_{lm}(-x, y, z) = Y_{lm}(\theta, \pi - \phi) = (-1)^m \bar{Y}_{lm}(\theta, \phi)$, where (θ, ϕ) are the angular coordinates of the point (x, y, z).

Unlike the previous subsections, the results in this subsection are specific to scalar spherical harmonics, although it seems reasonable to believe that the results could be generalized to spin-weighted spherical harmonics if needed.

Consider first the implications of symmetry on the metric G_{AB}. The symmetries of the spherical harmonics, summarized in Table 4.2, cause the imaginary part of all terms in the integral (4.29) to cancel in pairs of points on the sphere related by reflections through coordinate planes. The reason is that each of the four signs ($+1$, $(-1)^m$, $(-1)^l$, and $(-1)^{l+m}$) appears twice in Table 4.2, once for a term that is complex conjugated and once for a term that is not. The matrix is also sparse. By similar reasoning, for certain values of l and m, the terms in the integral (4.29) can cancel in sets of four. Both of these facts can be seen at once through a simple calculation. The idea is to break the integral into parts using the second and third columns of Table 4.2, and then to simplify using the last two columns. Considering first just the symmetries under reflection through the xy-plane and recalling the definition (4.29),

$$
\begin{aligned}
G_{AB} &= \oint \bar{Y}_{l_1 m_1}(\theta,\phi) Y_{l_2 m_2}(\theta,\phi) d^2\Omega && (4.36a)\\
&= \int_0^{2\pi}\int_0^{\pi/2} \bar{Y}_{l_1 m_1}(\theta,\phi) Y_{l_2 m_2}(\theta,\phi) d^2\Omega \\
&\quad + \int_0^{2\pi}\int_0^{\pi/2} \bar{Y}_{l_1 m_1}(\pi-\theta,\phi) Y_{l_2 m_2}(\pi-\theta,\phi) d^2\Omega && (4.36b)\\
&= [1+(-1)^{l_1+l_2}] \int_0^{2\pi}\int_0^{\pi/2} \bar{Y}_{l_1 m_1}(\theta,\phi) Y_{l_2 m_2}(\theta,\phi) d^2\Omega. && (4.36c)
\end{aligned}
$$

(I have suppressed the radial functions since they play no role here.) Repeating the procedure for reflections through the xz- and yz-planes shows that

$$
G_{AB} = 2\sigma_{m_1+m_2,l_1+l_2} \int_0^{\pi/2}\int_0^{\pi/2} \mathrm{Re}\left\{\bar{Y}_{l_1 m_1}(\theta,\phi) Y_{l_2 m_2}(\theta,\phi)\right\} d^2\Omega \tag{4.37}
$$

where

$$
\sigma_{m_1+m_2,l_1+l_2} \equiv 1 + (-1)^{m_1+m_2} + (-1)^{l_1+l_2} + (-1)^{m_1+m_2+l_1+l_2}. \tag{4.38}
$$

This proves that the matrix is real. In addition, the matrix element is zero by symmetry whenever

$$
\sigma_{m_1+m_2,l_1+l_2} = 0 \tag{4.39}
$$

which is true for 56 of the 81 matrix elements that exist when considering a fixed value of n and all values of l and m for $l \leq 2$. Of the remaining 25 matrix elements, 9, of course, are the diagonal elements that go to unity in the continuum limit. The exact break-down of which such elements must be zero by symmetry is summarized in Table 4.3. Knowing that the matrix is real-symmetric and sparse allows for a more efficient implementation of the algorithm in general. It is also extremely useful in analyzing the algorithm in the context of the second topic of this subsection, explicit grid symmetries.

When evolving initial data with known symmetries, it is very common to evolve only that part of the data that is unique. In such cases, an appropriate symmetry boundary condition is applied at some edges of the grid. This is, however, inconvenient for wave extraction since computing spherical harmonic components (by any method) requires integrating over the full sphere. If data with octant symmetry,

$m_1 + m_2$	$l_1 + l_2$	Number	Satisfies (4.39)
even	even	25	no
even	odd	16	yes
odd	even	20	yes
odd	odd	20	yes

Table 4.3: The table summarizes which entries of G_{AB} identically vanish because of the symmetries of the spherical harmonics under reflections through coordinate planes for all values of l and m with $l \leq 2$. (This is governed by equation (4.39).) Of the 81 possible matrix elements, only 25 have non-trivial values.

for example, is evolved only in a single octant, it is neither sufficient to apply the decomposition algorithm in that one octant nor to multiply the result of a single octant by 8 since the symmetry may forbid some modes as well as repeating them.

In principle the problem appears to be even more difficult for this particular decomposition method. Although the spherical harmonics have well defined symmetries under reflections, as summarized in Table 4.2, it is the adjoint harmonics that appear in (4.31). The adjoint harmonics, however, are constructed by contracting G^{AB} with the usual spherical harmonics, and this appears to mix different values of l and m. While this mixing does occur, the the matrix G_{AB} is sparse in just the right way to ensure that the adjoint harmonics have the same symmetries at the usual spherical harmonics.

A particular choice of the mapping $(n, l, m) \mapsto A$ makes this easiest to see. Specifically, considering all values of l and m with $l \leq 2$, there is a basis in which G_{AB} takes block diagonal form

$$(G_{AB}) = \begin{pmatrix} \Xi_1 & & & \\ & \Xi_2 & & \\ & & \Xi_3 & \\ & & & \Xi_4 \end{pmatrix} \tag{4.40}$$

with all unwritten entries identically zero by symmetry. In this expression Ξ_1 is a $4N \times 4N$ matrix over the basis functions $B_1 = \{Y_{n00}, Y_{n2,-2}, Y_{n20}, Y_{n22}\}$; Ξ_2 is an $N \times N$ matrix over the basis functions $B_2 = \{Y_{n10}\}$; Ξ_3 is a $2N \times 2N$ matrix over the basis functions $B_3 = \{Y_{n1,-1}, Y_{n11}\}$; and Ξ_4 is a $2N \times 2N$ matrix over the basis functions $B_4 = \{Y_{n2,-1}, Y_{n21}\}$. In this basis the matrix is block diagonal, so the inverse matrix

$$(G^{AB}) = \begin{pmatrix} \Xi_1^{-1} & & & \\ & \Xi_2^{-1} & & \\ & & \Xi_3^{-1} & \\ & & & \Xi_4^{-1} \end{pmatrix} \tag{4.41}$$

is also block diagonal and *the different basis sectors do not mix*. This last point is key. It implies that any particular adjoint harmonic is a linear combination of spherical harmonics from a single set B_k

$$Y^{nlm} = \sum_{Y_{n'l'm'} \in B_k} (\Xi_k)^{(nlm)(n'l'm')} Y_{n'l'm'} \tag{4.42}$$

where k is the index such that $Y_{nlm} \in B_k$. Because, in each set B_k, the parity of l and the parity of m is the same on each $Y_{nlm} \in B_k$, and because, in light of Table 4.2, it is the parity of l and m that determines the symmetries of Y_{nlm} under reflections through planes, every spherical harmonic in B_k for any fixed k has the same symmetries under reflections as any other spherical harmonic in B_k. This implies that the adjoint harmonics also share this symmetry under reflection.

4.2.5 Mesh refinement issues

Misner developed his method for extracting spherical harmonic components for the case of a uniform grid. Certain subtleties arise when the grid is not uniform that need to be considered in detail.

Algorithmically, the method remains largely unchanged, except that one needs to decide how to choose Δ when the extraction radius passes near or through a refinement boundary.[11] By experimenting with the method, I have found empirically

[11]One might argue that the simplest solution to this problem is to choose the refinement boundaries and the extraction radii such that there are no intersections. This solution is undesirable in practice since (1) fixed mesh refinement simulations are, in part, testbeds for *adaptive* mesh refinement simulations of the future, and in AMR such choices are not possible; and (2) for simulations based on the Paramesh package [61], as are the simulations presented here, the most efficient

that, even on FMR grids, it is safe to use the rule of thumb developed in Section 4.2.3. In particular, in the numerical simulations that follow, I choose $N = 2$ and $\Delta = 3h_{\max}/4$, where $h_{\max}$ is the grid spacing of the coarsest grid through which the shell could pass. (The details of the algorithm that makes this choice are in Section B.3.1.)

It should not be too surprising that this choice works even with FMR. The error analysis of Section 4.2.2 did not assume anything in particular about uniform grid spacing, except, implicitly, that the waves pass faithfully through the interfaces. Moreover, the reasoning that lead to my rule of thumb in Section 4.2.3 supports this choice. Taking $N = 2$ pushes the error term proportional to Δ to sub-leading order, so the exact size of Δ is not terribly important so long as there are enough zones in the shell to resolve the radial polynomials. Choosing Δ based on the coarse grid spacing means that there is at least as many points in the shell in the FMR run as there would be in a correspondingly spaced unigrid run.

What remains now is the implicit assumption in the analysis that waves pass faithfully through the mesh refinement boundaries. This, however, was already verified in detail in Ref. [30]. In what follows, I recreate those simulations as a testbed for all aspects of the wave extraction algorithm. I describe the details of those tests in Section 4.3.

4.3 Teukolsky Waves

Teukolsky waves are solutions to the linearize Einstein equations. These form an ideal test case for the methods considered here because the analytic solution is known for all times, which makes it easy to check simulation results, and because, for a linear wave, one need not wait for the wave to propagate to the wave zone to make a proper test.

box-in-box refinement scheme has little or no space for a shell of finite thickness to fit without intersecting refinement boundaries.

4.3.1 Analytic Even Parity Solution

Teukolsky appears to be the first person to write out even and odd parity wave solutions to the linearized Einstein equations [86]. I will use the pure $l = 2$, $m = 0$ (in a spin-weight -2 basis) solution as a test case in Section 4.3.2 after describing the solution in some detail in this subsection.

The general form of the even parity metric derived in Ref. [86]

$$\begin{aligned} ds^2 &= -dt^2 + (1 + Af_{rr})dr^2 + 2Bf_{rr}rdrd\theta + 2Bf_{r\phi}r\sin\theta drd\phi \\ &\quad + \left(1 + Cf^{(1)}_{\theta\theta} + Af^{(2)}_{\theta\theta}\right) r^2 d\theta^2 + 2(A-2C)f_{\theta\phi}r^2\sin\theta d\theta d\phi \\ &\quad + \left(1 + Cf^{(1)}_{\phi\phi} + Af^{(2)}_{\phi\phi}\right) r^2\sin^2\theta d\phi^2 \end{aligned} \tag{4.43}$$

is given in terms of the functions

$$A = 3\left(\frac{F^{(2)}}{r^3} + \frac{3F^{(1)}}{r^4} + \frac{3F}{r^5}\right) \tag{4.44a}$$

$$B = -\left(\frac{F^{(3)}}{r^2} + \frac{3F^{(2)}}{r^3} + \frac{6F^{(1)}}{r^4} + \frac{6F}{r^5}\right) \tag{4.44b}$$

$$C = \frac{1}{4}\left(\frac{F^{(4)}}{r} + \frac{2F^{(3)}}{r^2} + \frac{9F^{(2)}}{r^3} + \frac{21F^{(1)}}{r^4} + \frac{21F}{r^5}\right) \tag{4.44c}$$

and in terms of a free generating function $F = F(t - r)$. The notation

$$F^{(n)} = \left[\frac{d^nF(x)}{dx^n}\right]_{x=t-r} \tag{4.45}$$

denotes various derivatives. Taking F as a function of $t - r$ corresponds to outgoing waves. To generate an ingoing solution, change the argument of F to $t + r$, and change, in (4.44), the sign in front of all of the terms with *odd* numbers of derivatives.

The angular functions in the metric are

$$f_{rr} = 2 - 3\sin^2\theta \tag{4.46a}$$

$$f_{r\theta} = -3\sin\theta\cos\theta \tag{4.46b}$$

$$f_{r\phi} = 0 \tag{4.46c}$$

$$f^{(1)}_{\theta\theta} = 3\sin^2\theta \tag{4.46d}$$

$$f^{(2)}_{\theta\theta} = -1 \tag{4.46e}$$

$$f_{\theta\phi} = 0 \tag{4.46f}$$

$$f^{(1)}_{\phi\phi} = -f^{(1)}_{\theta\theta} \tag{4.46g}$$

$$f^{(2)}_{\phi\phi} = 3\sin^2\theta - 1 \tag{4.46h}$$

for the $l = 2$, $m = 0$ case.

I follow Choi el al. [30] in choosing

$$F(x) = \frac{ax}{\lambda^2} e^{-x^2/\lambda^2} \tag{4.47}$$

as the exact form of the generating function, where the free parameters a and λ represent the amplitude and the width of the wave respectively. The natural length unit in the problem is λ, and I consistently choose $a = 2 \times 10^{-6}\lambda$. Moreover I take an equal superposition of an ingoing wave and an outgoing wave, both centered at the origin, for the initial data. This particular choice has a moment of time symmetry that allows me to set $K_{ij} = 0$. Choosing this form for F gives a waveform with oscillations but of essentially compact support. This is ideal for testing codes with boundaries because it makes clear when the wave passes through those boundaries, and allows one to easily detect any reflections that occur due to poor interface conditions.

In order to use this analytic solution for tests of the wave extraction algorithm, I need to compute from it the analytical value of Ψ_4. The computation proceeds by applying the definition (4.3e) with the particular choice of tetrad (4.5). Of the twelve non-zero components of the Riemann tensor for this spacetime, only

$$R_{t\theta t\theta} = -\frac{1}{\sin^2\theta} R_{r\phi r\phi} = -\frac{3}{2} r^2 \sin^2\theta \frac{\partial^2 C}{\partial t^2} + \frac{1}{2} r^2 \frac{\partial^2 A}{\partial t^2} \tag{4.48a}$$

Label	Unigrid		FMR		
	dx	$x_{\max}$	dx_{fine}	$x_{\max}$	Interfaces
High	1/12	8	1/48	32	2, 4, 8, 16
Medium	1/6	8	1/24	32	2, 4, 8, 16
Low	1/3	8	1/12	32	2, 4, 8, 16

Table 4.4: The grid parameters used for the various Teukolsky wave test runs. The simulation's outer boundary is at $x_{\max}$, and the value of dx_{fine} is the value of the grid spacing at the finest FMR level. Other FMR levels are factors of two coarser. All numbers are given in units of the wavelength λ.

$$R_{t\phi t\phi} = -\sin^2\theta R_{r\theta r\theta} = \frac{3}{2}r^2\sin^4\theta\left(\frac{\partial^2 C}{\partial t^2} - \frac{\partial^2 A}{\partial t^2}\right) + \frac{1}{2}r^2\sin^2\theta\frac{\partial^2 A}{\partial t^2} \tag{4.48b}$$

$$R_{t\theta r\theta} = -\frac{1}{\sin^2\theta}R_{t\phi r\phi} = -\frac{1}{8}r^3\sin^2\theta\left(3\frac{\partial^3 B}{\partial t^3} + \frac{\partial^3 A}{\partial t^3}\right) \tag{4.48c}$$

contribute to the sum. (In particular, this means that, in this spacetime, Ψ_4 is real.) The result

$$\Psi_4 = \frac{\sin^2\theta}{16}\left[-12\frac{\partial^2 C}{\partial t^2} + 6\frac{\partial^2 A}{\partial t^2} + r\left(3\frac{\partial^3 B}{\partial t^3} + \frac{\partial^3 A}{\partial t^3}\right)\right] \tag{4.49}$$

is a pure mode of ${}_{-2}Y_{20}$ (cf. (4.18)). The third time derivatives conveniently express the particular mix of second time and space derivatives that otherwise would have appeared in (4.49).

4.3.2 Numerical Results

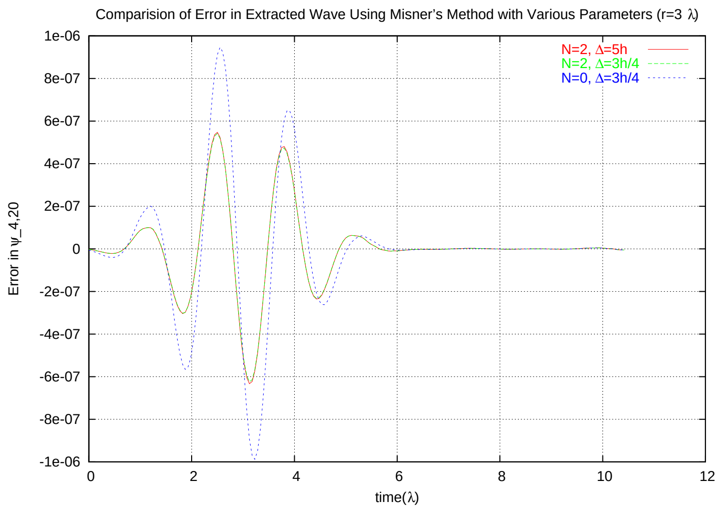
In order to test aspects both of the wave extraction method and the effects of FMR interfaces on wave extraction, I ran two classes of test cases. Each class consisted of three runs with different spatial resolutions. The runs of the first class were set on a uniform grid, while the runs of the second class were set on fixed mesh refined grids. The spatial parameters for the runs are summarized in Table 4.4. The time step was chosen to be $dt = dx/4$ in unigrid and $dt = dx_{\text{fine}}/4$ in FMR simulations.

The low resolution simulations, especially in the unigrid case, are noticeably coarse. This choice, driven primarily by hardware considerations, is sufficient for my purposes here, although in the cases of the coarsest simulations it is only marginally so. I should mention, so that it is clear why the unigrid simulations are run at such low resolutions compared to the FMR simulations, that the computational burden, in terms of memory and execution time, for the highest resolution unigrid run exceeds the computational burden of the highest resolution FMR run. This is true, despite the resolution being a factor of four smaller and the outer boundary being a factor of four closer, because the FMR simulation is much more efficient.

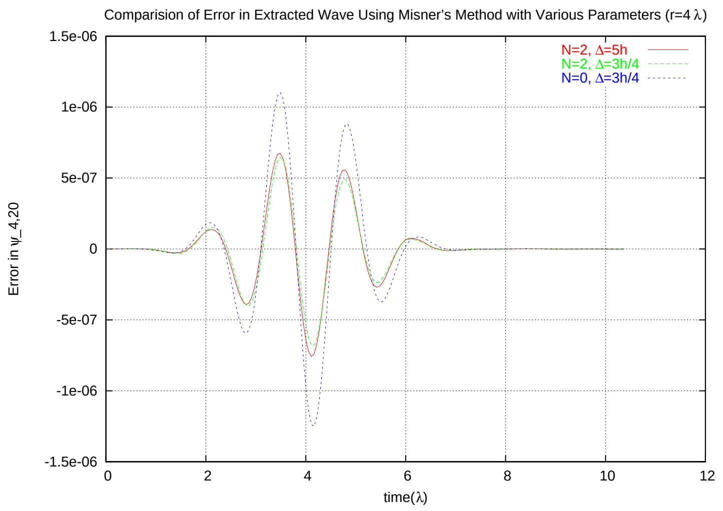
Before proceeding further, I need now to justify my choices of the parameters N and Δ to the Misner algorithm. As noted in Section 4.2.3, I have chosen here to use $N = 2$ and $\Delta = 3h/4$ for the simulations presented below. These choices are consistent with theoretical considerations, but are further justified empirically by Figure 4.2, which shows the error in waves extracted with three different parameter sets. When $N = 2$, it appears that there is little difference in the error as Δ varies (although there is arguably a slight increase in error for the larger Δ, consistent with the analysis in Section 4.2.2). Changing to $N = 0$ while keeping Δ fixed, however, leads to a considerably larger numerical error. (Keep in mind that only even values of N are allowed.) Based on this evidence, I now specialize strictly to the parameter choices $N = 2$ and $\Delta = 3h/4$ for the rest of the dissertation. In addition, I choose to compute weights and (scalar) spherical harmonics for all values of l and m, with $l \leq 2$.[12]

Having fixed both the grid and the wave extraction parameters, let me comment on how they relate to each other. To better visualize how the FMR boundaries intersect with the extraction spheres, I have found it useful to create an *extraction map*. Figure 4.3 shows extraction maps for the coarse FMR run, and Figure 4.4 shows extraction maps for the medium FMR run. As indicated by the figures, extraction radii were located at 3λ, 4λ, 5λ, and 6λ. In addition there was an extraction radius at 7λ for which no extraction map is shown. Each map is labeled by a triple of parameters (N, R, Δ), which indicate, respectively, the number of (cell-centered)

[12]In a single set of runs, I reproduced the highest resolution FMR runs computing all values of l and m with $l \leq 3$. I verified in this case that all scalar spherical harmonic components except for the (0,0) and (2,0) components were zero to extremely high precision.



(a)



(b)

Figure 4.2: The two panels show errors in the numerically extracted wave in high resolution, FMR simulations with three different wave extraction parameter sets. There is little difference between different shell thicknesses so long as $N = 2$. There is a large increase in error when the parameter N is reduced.

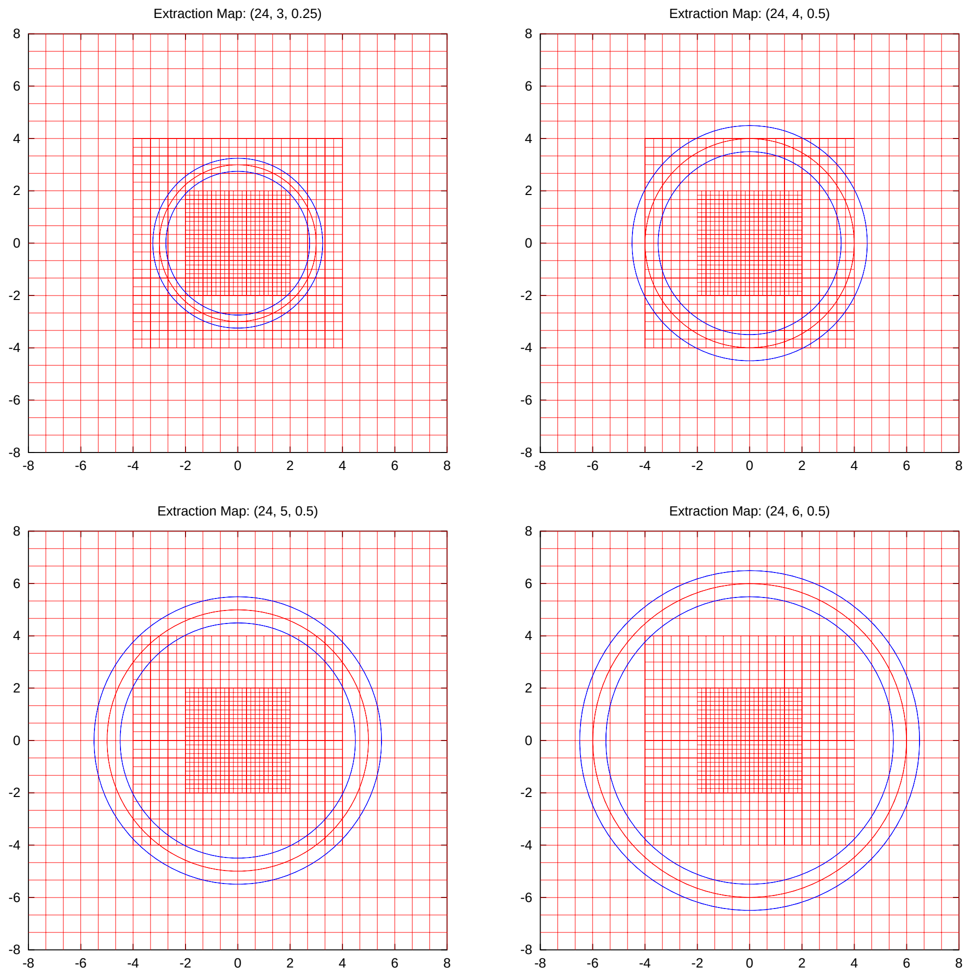

Extraction Map: (24, 3, 0.25)
Extraction Map: (24, 4, 0.5)
Extraction Map: (24, 5, 0.5)
Extraction Map: (24, 6, 0.5)
-8
-6
-4
-2
0
2
4
6
8

Figure 4.3: Extraction maps for four extraction radii in the coarse resolution Teukolsky wave FMR run. The mesh size indicates the grid spacing in the various refinement regions, while the three circles mark the extraction radius and the boundaries of the shell used in the spherical harmonic extraction algorithm. The labels at the top include a triple (N, R, Δ), where N is the number of (cell-centered) grid points across one coordinate direction in the coarsest region, R is the extraction radius, and Δ is the half-width of the shell. Additional refinement regions, not shown here, surround the entire map.

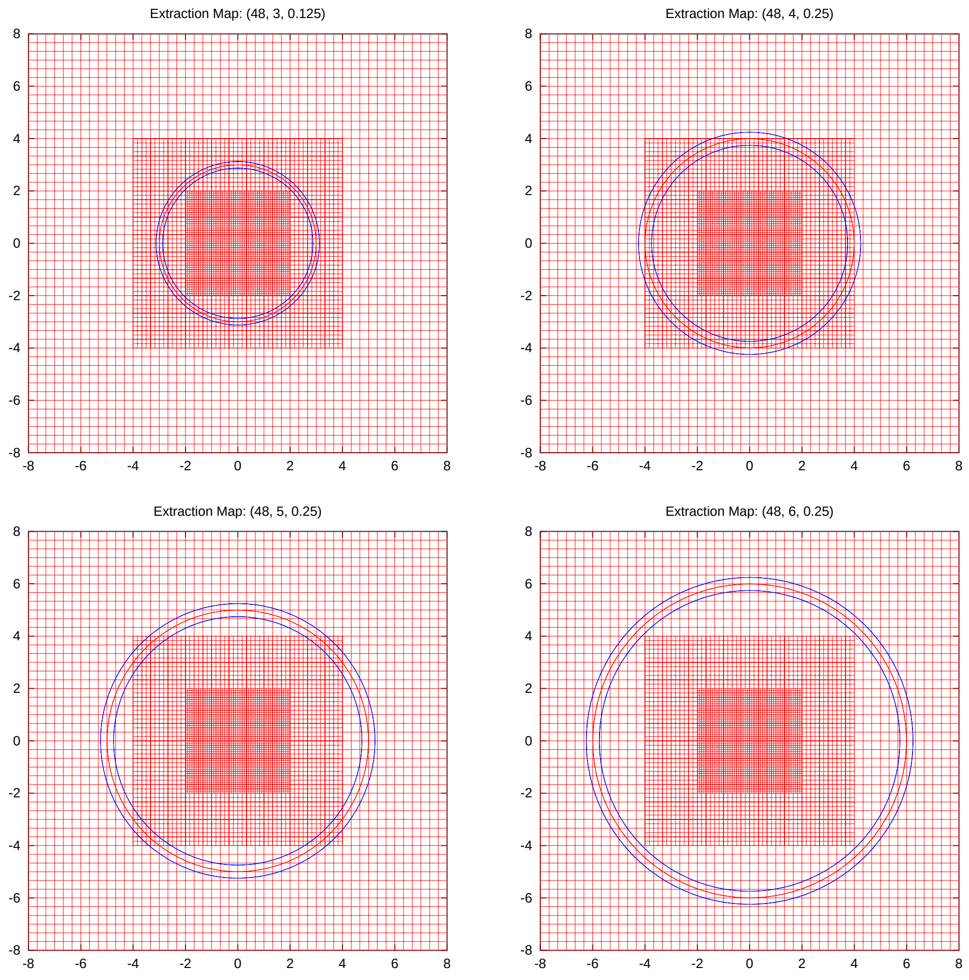
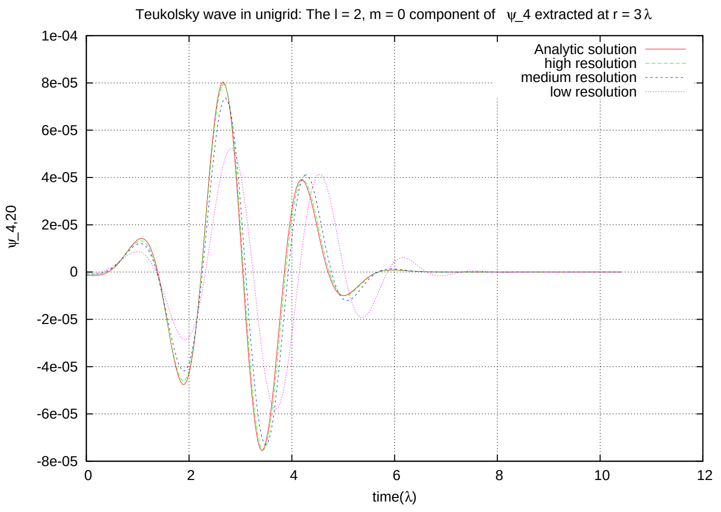

Extraction Map: (48, 3, 0.125)
Extraction Map: (48, 4, 0.25)
Extraction Map: (48, 5, 0.25)
Extraction Map: (48, 6, 0.25)

Figure 4.4: Extraction maps for four extraction radii in the medium resolution Teukolsky wave FMR run. The mesh size indicates the grid spacing in the various refinement regions, while the three circles mark the extraction radius and the boundaries of the shell used in the spherical harmonic extraction algorithm. The labels at the top include a triple (N, R, Δ), where N is the number of (cell-centered) grid points across one coordinate direction in the coarsest region, R is the extraction radius, and Δ is the half-width of the shell. Additional refinement regions, not shown here, surround the entire map.

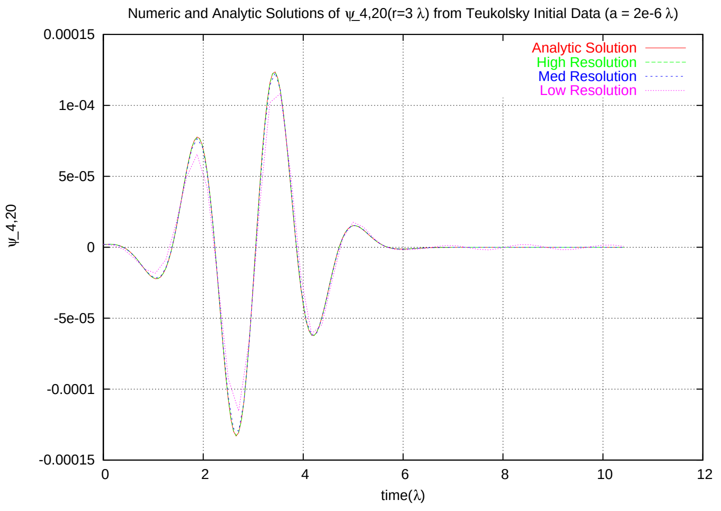
points across one coordinate direction in each refinement level, the extraction radius, and the half-width of the extraction shell. The mesh accurately portrays the grid refinement in each region, and the extraction region is denoted by three circular lines; the middle line is the extraction radius, and the two surrounding lines are the edges of the extraction shell. Note that the extraction shells generically pass through mesh refinement boundaries. One should also bear in mind that these two-dimensional slices depict the extraction sphere at its largest cross section and that the refinement regions are cubical. That means, for example, that, in three dimensions, the corners of the innermost refinement cube poke through the sphere represented by map $(48, 3, 0.125)$ in Figure 4.4, even though, in the two dimensional picture, it appears that the extraction sphere might be in a single resolution region.

Having understood the grid structure in the various runs, turn now to the results. Before looking at each family of runs individually, it is worth making a direct comparison between the raw results. Figure 4.5 shows analytic and numeric solutions to the Teukolsky wave equation at $r = 3\lambda$ for the entire family of unigrid runs (Panel (a)), and the entire family of FMR runs (Panel(b)). Both families appear to show agreement with the analytic result at the highest resolutions, and appear to converge to the analytic solution. In addition, the FMR runs appear, consistent with the fact that they were run with higher resolution in the extraction region, to more accurately represent the exact answer. These observations are quantified and expanded below.

In the case of uniform grids, the analysis in Section 4.2.2 indicates that the numerical value of $\Psi_{4,20}$ extracted from a second order accurate simulation should be second order accurate. This is indeed the case, as seen in Figure 4.6. In the figure, the errors in the numerical solution as compared to the analytic solution have been scaled by appropriate factors of two so that, for perfect second order convergence, the lines would lie over each other. This is in fact the case. The lowest resolution run shows signs of a phase error and dissipation that is common for poorly resolved waves propagating on a numerical grid, but the higher resolution simulations are clearly passing into the convergent regime. This validates, for the first time in a live simulation, the spherical harmonic decomposition method, and validates, for the first



(a)



(b)

Figure 4.5: Analytic and numeric solutions for the $l = 2$, $m = 0$ component of a Teukolsky wave at three resolutions. No mesh refinement was used for any of the runs shown in Panel (a), and the extraction radius is $r = 3\lambda$. Mesh refinement was used for the runs in Panel (b). The extraction radius is also at $r = 3\lambda$, which passed through a cubic mesh refinement boundary of side length 2λ. (Additional refinement boundaries exist beyond the extraction radius.) Note the agreement between the high resolution runs and the analytic solution.

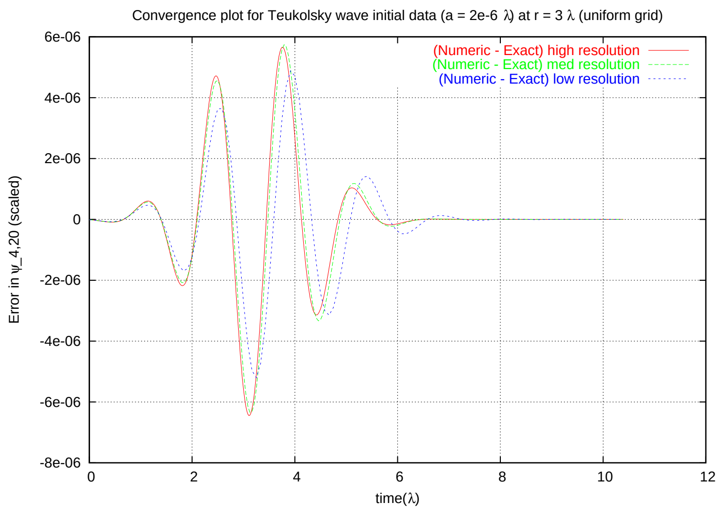
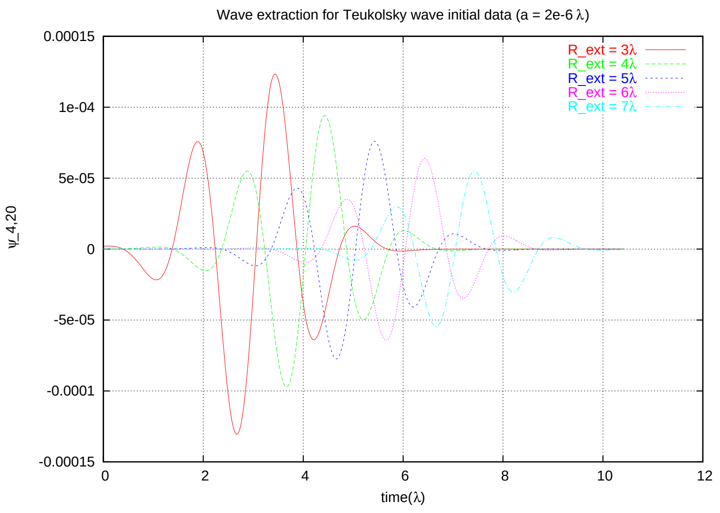


Figure 4.6: A convergence plot for the $l = 2$, $m = 0$ component of Ψ_4 extracted from unigrid simulations of a Teukolsky wave at radius $r = 3\lambda$, where λ is the width of the wave in the initial data. The lowest resolution suffers from a phase error, which is to be expected from a low resolution simulation.

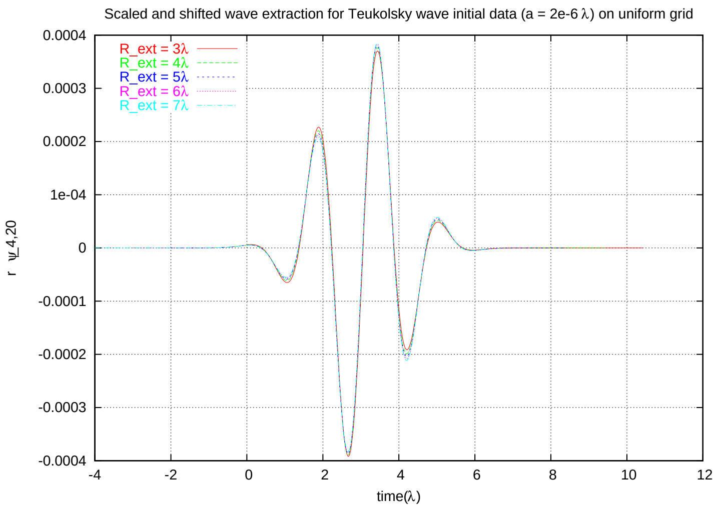
time ever, the error estimates made in Section 4.2.2. It is also worth noting that, comparing Panel (a) of Figure 4.5 with Figure 4.6 at the highest resolutions used, the error in the waveforms is on the order of a few percent, which is a good result considering resolution.

The overall quality of the waveforms extracted from the simulation can also be verified by comparing results extracted at several distinct radii. Because the dominant r dependence of Ψ_4 goes like r^{-1}, the easiest way to compare multiple extractions is to shift the waveform in space and scale it by r so that the different waveforms should lie nearly on top of each other; this agreement will not be perfect even in the analytic solution since Ψ_4 also has terms that scale with higher negative powers of r. Figure 4.7 shows both the raw $\Psi_{4,20}$ data (panel (a)) extracted at 3λ, 4λ, 5λ, 6λ, and 7λ, and that same data shifted to $r = 3\lambda$ and scaled up by r. It is clear from panel (b) that the wave is faithfully propagated and extracted at all radii considered.

The numerical results for the FMR family, shown at $r = 3\lambda$ in Panel (b) of Figure 4.5, are better approximations to the analytic result than the unigrid runs shown in Panel (a). Because the FMR grids are more memory efficient, higher resolutions



(a)



(b)

Figure 4.7: Panel (a) shows the $l = 2$, $m = 0$ component of Ψ_4 extracted from a unigrid run at various extraction radii. Panel (b) shows the same data, shifted to $r = 3\lambda$ and scaled by r so that the waveforms should lie nearly on top of each other.

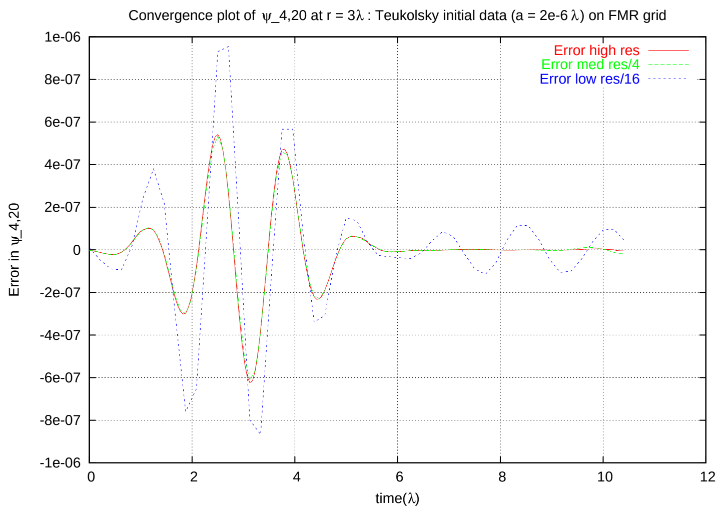
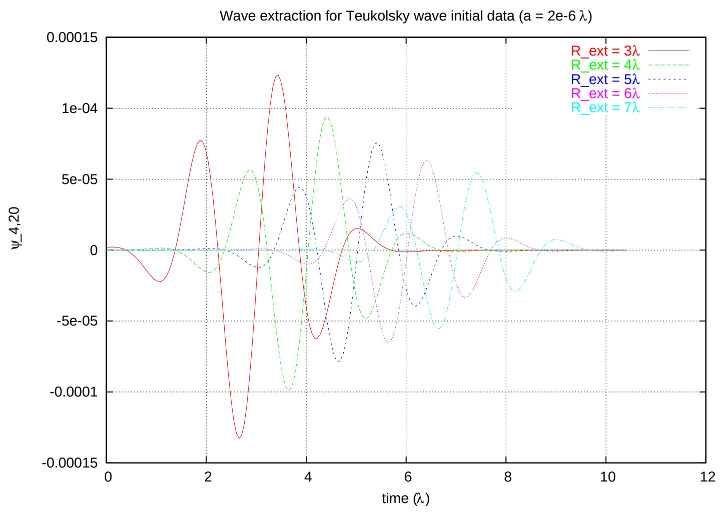


Figure 4.8: A convergence plot for the value of $\Psi_{4,20}$ at $r = 3\lambda$ extracted in simulations at three different resolutions. The lowest resolution result is clearly not in the convergent regime, but the two highest resolutions show excellent convergence.

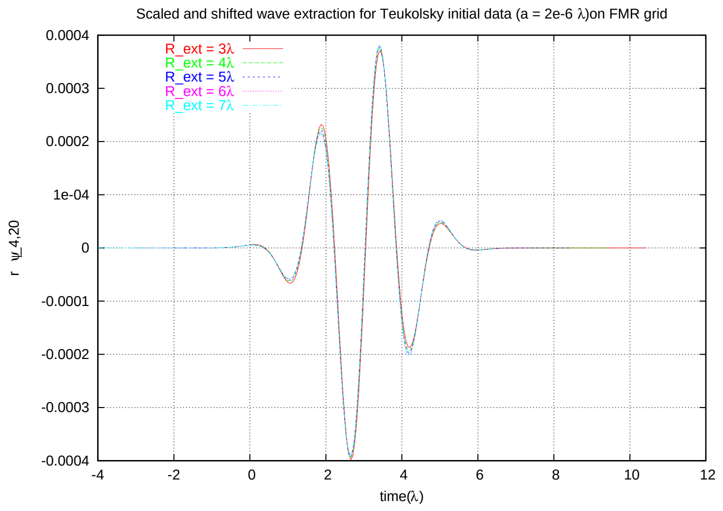
are achievable, and because both the guardcell filling and extraction algorithms are well behaved in the presence of FMR boundaries, this higher resolution leads to more accurate waveforms. This is apparent in Figure 4.8, which shows the errors in the numeric solutions scaled by powers of two appropriate to second order convergence. The scaled error curves for the two highest resolutions overlay because the simulation is in fact converging at second order. The lowest resolution has not reached the convergent regime, but its absolute errors are still quite small.[13] By comparing the peak error in the finest resolution curve in Figure 4.8 to the amplitude of the extracted wave in Panel (b) of Figure 4.5, it is clear that the error in the waveform at these resolutions is on the order of a few tenths of a percent. The validity of these waveforms is further confirmed by looking at the scaled and shifted waveforms extracted at several radii. Figure 4.9 shows both the raw data extracted at several radii (Panel (a)), and that same data scaled and shifted to $r = 3\lambda$.

In the end, these experiments with the Teukolsky wave indicate that Misner's

[13]Compare Figure 4.8 with Figure 4.6. The lowest resolution in the FMR run has resolutions in its central regions comparable to the two lowest resolution unigrid runs. See Table 4.4 for the exact parameters.



(a)



(b)

Figure 4.9: Panel (a) shows the $l = 2$, $m = 0$ component of Ψ_4 extracted from an FMR run at various extraction radii. Panel (b) shows the same data, shifted and scaled by r so that the waveforms should lie nearly on top of each other.

method for extracting spherical harmonics components works in an actual application to general relativity, and that the error analysis of that method, presented for the first time here is also born out in numerical results. The experiments show further that the presence of FMR interfaces neither degrades the performance of the algorithm nor spoils the propagation of the wave itself.

These results represent a crucial step toward extracting gravitational radiation signal from more realistic simulations. The technology developed here is directly applicable to black hole spacetimes without change, provided that the simulation of the black hole itself is both stable and reliable. Because there has been recent progress in the art of constructing robust black hole simulations, there is good reason to hope that black hole simulations of the near future will apply the techniques developed here to extract signals from distorted black hole or binary black hole simulations.

4.4 Summary and Future Work

In this chapter I outlined the basic mathematical tools used to describe gravitational radiation, the Newman-Penrose formalism. Within that formalism, I described the need for and the definition of spin-weighted spherical harmonics, which are the generalization of the usual scalar spherical harmonics appropriate for describing radiation fields in general relativity. I also presented one particular algorithm due to Misner for computing spin-weighted spherical harmonics from data numerically evolved on a cubic grid.

Working from this background, I presented a detailed error analysis of Misner's algorithm and worked out, in the case of the scalar spherical harmonics, all of the details necessary to implement the algorithm on a cubic grid in which reflection symmetries are explicitly enforced by boundary conditions. Also, related to the issue of symmetry, I showed that a particular matrix, which plays a key role in Misner's algorithm, is real-symmetric and sparse, rather than complex-Hermitian and dense as would naively be thought. This can lead to significant savings in computation, especially if the method is pushed to higher order accuracy, used to compute spherical harmonics with higher values of l, or used in a simulation in which the weights must be recalculated during the simulation.[14]

This work was tested in the context of a linearized solution to the Einstein equations with a known spherical harmonic expansion. The theoretical work of the earlier sections of the chapter were tested both in the case of uniform grids and grids with fixed mesh refinement interfaces. Waves extracted and decomposed into spherical harmonic components, on both types of grids, matched the analytic solution to high degree of accuracy. The error models presented here were demonstrated to be correct.

The benefits of using FMR are also apparent. The results of this chapter demonstrate that radiation quantities (i.e. Ψ_4) propagate without significant distortions through properly treated mesh refinement boundaries. This builds on the results of Ref. [30], which showed that metric quantities also propagate through mesh refinement boundaries. Moreover the results of this chapter demonstrate that it is possible to compute the spherical harmonic decomposition of radiation quantities even when the extraction sphere intersects mesh refinement boundaries. This is a significant result because the FMR simulations are much more efficient to run on the computer, allowing higher resolutions in the central regions of domain and pushing the physical boundary (for which there is no known, physically consistent boundary condition to apply in the strong field region) farther out. This simultaneously improves the accuracy of the simulation in the "source" region (looking forward toward more realistic simulations in which compact objects generate the gravitational radiation), while pushing the physical boundary far enough away that any incoming radiation generated by the boundary condition does not have time to propagate to the extraction radius.[15]

[14]Although I only need to compute the weights at the beginning of a simulation, in other applications the weights might need to be recomputed many times. This would occur if the extraction radius changed during the course of a simulation or if the grid structure changed in a simulation using *adaptive* mesh refinement.

[15]The lower resolution in the outer regions also tends to damp any wave signal, and, in addition, when the boundary is far enough away, it is possible to impose reasonable boundary conditions. Both of these facts conspire to reduce the effects of the outer boundary conditions on the simulation.

On the other hand, it is also clear that FMR is not a fix-all for resolution problems. The wave must be properly resolved in all regions between the source and the extraction radius in order to extract a proper waveform. This is apparent in Figure 4.8, which shows that the error in the lowest resolution simulation is not yet converging. In that low resolution run, the innermost refinement region has a resolution of $\lambda/12$ and the second refinement region, which contains the extraction shell, has a resolution of just $\lambda/6$. Most people would not expect a unigrid simulation with less than ten points per wavelength to converge, and, indeed, one should not expect to be able to successfully extract a waveform in a mesh refined region of such low resolution either. In more realistic simulations, in which the source is a strong-field interaction (a black hole binary, for example), this could pose problems even for an FMR simulation since, in those cases, the extraction radius will be farther from the center of the grid at a distance corresponding to the radiation zone. An FMR simulation will need to maintain sufficient resolution in all zones out to the extraction radius. Of course the FMR simulation is still more efficient than a unigrid run that maintains the highest resolution all the way to the wave zone,[16] and AMR could solve this problem by localizing resolution on the wave packets as they travel outward.

These results provide a crucial step toward extracting gravitational wave signals from more physically and astrophysically relevant simulations. They can be extended in the near future to studies of distorted black hole initial data. These studies are of interest to gravitational wave detector experiments because, after colliding, black holes form a single, highly distorted black hole, which continues to radiate gravitational energy. Some of these signals may be detectable. Even if the application to strongly distorted black holes is not immediately possible, which may be the case since they have not been well studied numerically, applications to weakly distorted

[16]Keep in mind that in a simulation with an astrophysical source, say a binary black hole system for example, the characteristic size of the source is of the order of the system mass M, whereas the radiation emitted is of the order $10M$–$100M$. That means that the source region needs to have grid spacings of order $M/10$ at maximum, whereas the wave zone only needs grid spacings of order M. (Of course higher resolutions, in both regions, are desirable.) In a unigrid simulation one would need to keep the highest resolution through the whole grid.

black holes will also have research benefits. Some people in the community (primarily the Lazarus project [10]) have studied the possibility of interfacing fully non-linear numerical codes to codes that evolve only linear perturbations. The ability to make direct comparisons between results generated from a non-linear code and results generated by a combination of non-linear evolution followed by an evolution of linearized perturbations would be a great aid in evaluating the usefulness of the mixed approach. Very likely it would lead to improvements in both methods.

Brill waves [20] are a likely test case for the ideas and methods presented in this chapter, which could form an intermediate step between the linearized waves treated here and distorted black holes. While black holes introduce additional technical problems related to representing singularities inherent in the solution on the computer, Brill waves are a direct generalization of the work presented in this chapter. In addition, distorted black hole initial data is frequently generated by superposing a Brill wave on an undistorted black hole solution, and relaxing (by solving an elliptic equation) that initial guess to a solution of the constraints (see, e.g., Ref. [16]). In order to generate Brill wave initial data in any case, one needs to solve a scalar elliptic equation, but the Goddard group is fortunate to have access to a numerical elliptic solver that is also layered over the Paramesh package and which is able to solve the equation on a mesh refined grid compatible with the Hahndol evolution code [22]. This technical problem is no longer an obstacle.

The ability to integrate radiation quantities over a sphere will also play a role in our attempts to study "kicks" generated by black hole collisions. When two compact objects with non-equal masses collide, the gravitational radiation field emitted will not be symmetric. This means that the black hole formed by the collision will have non-zero linear momentum in the center of mass frame of the two initial objects. (This is necessary because the gravitational waves carry linear momentum in a preferred direction.) Order of magnitude estimates suggest that the kick velocity of the final object is bounded by $v < 10^{-4}c = 300$ km s^{-1}. If kick velocities actually approach this bound, they could be sufficient to eject a merged black hole from the center of a galaxy, thereby preventing a large central black hole from forming there via hierarchical mergers [75]. In order to study this in a numerical simulation, it is necessary to record and integrate in time the amount of radiation passing through some large sphere. Any net momentum flux passing through the sphere must be compensated by a change in momentum of the final black hole. The calculation requires extracting the Y_{1m} (scalar) spherical harmonic components of the time-integrated radiation field, which is exactly what we now know how to do.

In terms of future algorithm development, the primary impediment to pushing the algorithm to higher order is the simple weighting scheme used to compute the volume integral. It would be nice to develop a more accurate way to compute these integrals and incorporate it into the code. Although few people have a code that is only first order accurate, there are currently multiple efforts to introduce codes that converge at higher than second order. It would not be acceptable to use a second order accurate analysis algorithm on a fourth order accurate simulation. Even among second order codes, it would be preferable to see the order of accuracy improved because it is preferable that the numerical errors from the analysis step be of higher order than the numerical errors in the data.

Chapter 5

Conclusions and Discussion

This dissertation has focused on the role of the constraints in numerical evolutions, and on the mathematical and algorithmic issues associated with computing gravitational waveforms from numerically evolved data. Although the specific problems studied here have been simpler than those of interest to the observational community, they have provided valuable testbeds for new algorithms. In the case of the constraint driver algorithm described in Chapter 3, more work is needed to show that the method will be robust in more realistic situations, while the wave extraction technology developed in Chapter 4 is already implemented in a code that can, in principle, handle more astrophysically interesting problems. Taking a broader view, I noted in the introduction, and wish to reiterate here, that much of the original work presented in this dissertation has applications beyond numerical relativity. Constraint driver terms can be applied to any set of partial differential equations subject to a constraint, and Misner's algorithm is applicable to any problem for which spherical harmonic components should be extracted from data represented on a cubic lattice.

On the issue of the constraints, the dissertation presents a novel prescription for incorporating the constraint equations of the theory into the evolution equations. I showed in the text, for the cases studied here, that with this particular modification, arbitrary initial data will evolve into data that satisfies the constraints of the theory. This is true at the analytic level. In four separate linear test cases, two cases of the Maxwell equations in three dimensions and two cases of the linearized Einstein equations in one dimension, numerical simulations showed behavior consistent with this prediction. More importantly, I showed that, for initially constrained data, numerical constraint violations are suppress in a way that decreases or leaves unchanged the total solution error. If these results generalize to non-linear equations, this method will play an important role in future simulations throughout numerical relativity.

The work that I presented on this topic in the dissertation was partially based on results that I had already published, which showed that the constraint driver terms worked for the Maxwell equations. In this dissertation, I have added to that by successfully applying the technique to the linearized Einstein equations. The application to Einstein differed from the application to Maxwell in that the correction terms for the Einstein equations contain higher order derivatives. In the Maxwell equations, the correction terms added derivatives of the same order as those already in the equations. Although adding fourth derivatives may have seemed, a priori, to risk introducing high frequency instabilities, I now have numerical evidence that they do not appear to cause any problems in the bulk of the simulation. (Because I used periodic boundary conditions in my test problem, I have avoided, so far, any potential complications the fourth derivatives create at the boundaries.)

In my original paper on this subject, I identified two features of the corrections to the Einstein equations that differed from the Maxwell corrections. The first was higher order derivatives, which I just discussed. The second, which remains untested is the non-linear nature of the corrections to Einstein. It appears, in fact, that the correction terms are not even quasi-linear. While this is not a guaranteed problem, there is little mathematical theory available for partial differential systems that are not quasi-linear. This remains one of the most interesting aspects of the method to test, and it is a problem that I look forward to studying in the near future.

On the topic of wave extraction, I made a detailed study of Misner's algorithm for extracting spherical harmonic components from data represented on a cubic lattice. In particular, I made a detailed error analysis of the method as a function of all of its parameters, and I used the symmetries of the spherical harmonic functions to determine an optimal way of applying the method on grids in which symmetries in the data are imposed by boundary conditions. (Meaning, for example, that only one octant of the sphere is numerically evolved for data with octant symmetry.) I implemented the algorithm, and used the code in conjunction with an existing code designed to evolve the fully non-linear, three dimensional Einstein equations. The numerical results from these tests confirmed my error estimates, both in the case of a grid with uniform resolution and in the case of fixed mesh refinement.

Because my implementation of the algorithm is already working with a generic Einstein solver, I am already in an excellent position to exploit this new technology on more interesting problems. From an astrophysical point of view, one would like to study the gravitational radiation emitted from distorted black holes since a distorted black hole is the end product of a binary black hole collision. This is certainly something that I will try in the near future. Another problem of pedagogical value would be Brill wave spacetimes. Both cases would test the technology in situations that are in the non-linear regime of the equations, but, while there may be technical issues associated with distorted black hole spacetimes, which have not been very well studied, the Brill wave should be immediately accessible. (The technical differences in the problem arise because the black hole has a physical singularity, which makes the problem more challenging even without attempting to extract a meaningful waveform.)

Future work on the wave extraction algorithm will almost certainly be centered on making it more accurate. The error analysis that I provided in the text indicates exactly how to modify the various parameters of the algorithm to achieve any desired level of convergence, but this is currently of limited value. The dominant error in the method at present is in the approximation to the volume integral over a thin shell. To really push the method to higher accuracy, a better integration scheme will be required.

In a wider context, while there is no doubt that while the dissertation makes contributions both to the overall understanding of the role of the constraints in free evolution and to the development of techniques for successfully computing gravitational waveforms from simulated data, there is also no doubt that these are distinct contributions to the field. One goal of future work will certainly be to combine both advances into a single framework, so that they can both be applied simultaneously. To a large extent I have mapped the road to achieving this already. I need to demonstrate that the constraint driver terms will continue to perform as desired when the equations are non-linear, and the I need to compute and implement the corrections in three dimensions. (Although for the relatively simple problems presented in the text computing the corrections was nearly trivial, computing the corrections for the full Einstein equations in three dimensions promises to be a tedious process.) Once I have such an implementation, I will be able to include the correction terms in the evolution code from which I generate my radiation quantities.

The techniques developed here must also be merged with advances from other areas of numerical relativity. When simulating black hole spacetimes, for example, the singularity inherent to the physical solution must somehow be modeled. One of the leading techniques for treating the singularity is the excision method, which was discussed only briefly in the text. In this method the interior of the black hole is simply not evolved. While this presents no theoretical problem since the black hole interior is causally disconnected from the exterior, it presents a computational problem because it introduces an interior boundary in the simulation domain. Several groups have worked on ways to stably apply boundary conditions at the excision boundary, but it is not immediately clear that these techniques will be compatible with the constraint driver terms that I have proposed. The fourth derivatives present in the constraint driver corrections would surely make the analysis more complicated, and this needs to be studied. Although there is no clear impediment to pushing the techniques that I have developed in the dissertation to more realistic situations, issues like this are apparent and will need to be studied in the future.

Appendix A

Fixed Mesh Refinement

This appendix provides some details associated with fixed mesh refinement as implemented in the Hahndol code, which was used to generate some of the data presented in this dissertation. Fixed mesh refinement, although well developed in other computational fields, remains a novel technology in numerical relativity. Because numerical relativity deals with non-linear equations, with high order derivatives, and which do not appear to admit a manifestly conservative formalism, not all of the experiences of other fields is immediately applicable to fixed mesh refinement in numerical relativity. These problems are highlighted below. The discussion on guard cells comes, in some cases directly, from Ref. [55].

The Hahndol code [30, 55] is layered over the Paramesh libraries [61], which create and manipulate the data structures required for mesh refinement and handle most of the parallelization of the code. I will not attempt to explain in full the details of the Paramesh package, nor of Hahndol itself, but wish to use them as examples in a sketch of some key points that will help the reader, especially a reader unfamiliar with mesh refinement techniques, to understand the computational science and mathematical issues involved in constructing proper interface conditions.

We use a "box-in-box" refinement scheme. This is illustrated in two dimensions in Figure 4.3, which shows relatively fine grids near the origin surrounded by coarser grids. Adjacent grids always differ in grid spacing by a factor of two. There is no refinement in time, which means that all grids, independent of spatial resolution are updated together with a time step that respects the Courant stability condition on the finest grid. This differs from the Berger-Oliger scheme [15] that is common in other fields, as well as the modified Berger-Oliger schemed first introduced by the Carpet collaboration in numerical relativity [74]. In these schemes each grid is updated according to its own Courant factor. While the Berger-Oliger scheme is a factor of four more efficient, our numerical algorithm is easier to construct, and is therefore, arguably, an easier test case.

We use a cell-centered scheme for representing the data. This means that, in Figure 4.3 for example, we store data not at the intersection of the grid lines, but at the center of the cells formed by the crossing grid lines. In a unigrid simulation this might make little difference, but in cases with multiple refinement regions, this has a practical effect: the data on two neighboring grids is not aligned in any spatial direction. Considering that our finite differencing stencils, like the first derivative approximation

$$f'(x_i) \approx \frac{f(x_{i+1}) - f(x_{i-1})}{2\Delta x} \tag{A.1}$$

for example, extend at least one grid point in each direction, this means that we need to "create" data in order to apply those stencils at a refinement boundary. This extra data is stored at fictional points called guard cells, which are filled using some sort of interpolation.

The primary problem in introducing multiple grids into a computation simulation is handling the interface conditions between the refinement regions. In physical terms, the different refinement regions effectively have different indices of refraction, so that waves incident on them may reflect and refract. The interface conditions, which means the interpolation scheme used to fill the guard cells, applied at the boundaries between refinement regions must be constructed carefully to ensure that such effects are minimized, and also to ensure that the overall convergence order of the code is maintained.

As a general rule, guard cells at mesh refinement boundaries should be filled to third order accuracy for an evolution code such as ours to maintain overall second order accuracy.[1] We point out that this is not the result one would expect from a

[1]In our terminology the "order of accuracy" refers to the order of errors in the grid spacing.

simple counting of powers of $\Delta x \sim \Delta t$. For example, Eqs. (2.15) contain second spatial derivative terms on the right hand sides. Using standard $\mathcal{O}(\Delta x^2)$ centered spatial differences this gives, for any field f, $f''_j = (f_{j+1} - 2f_j + f_{j-1})/\Delta x^2$, where primes denote a spatial derivative and, for simplicity, we are looking only at the x-direction. Suppose that this term includes a guard cell, say f_{j+1}, that is computed to n^{th} order accuracy, $f_{j+1} \to f_{j+1} + \mathcal{O}(\Delta x^n)$. With this, the second derivative term becomes $f''_j \to f''_j + \mathcal{O}(\Delta x^{n-2})$. Maintaining overall $\mathcal{O}(\Delta x^2)$ accuracy thus appears to require filling the guard cells to fourth order accuracy. However, as we discuss below, third order accurate guard cell filling is sufficient to produce second order accurate evolutions; cf. [29, 49].

The current version of our code uses a third order guard cell filling scheme that is now included with the standard Paramesh package. This guard cell filling proceeds in three steps.

The first step is a restriction operation in which interior fine grid cells are used to fill the interior grid cells of the underlying "parent" grid. The parent grid is a grid that covers the same domain as the fine grid but has twice the grid spacing. The restriction operation is depicted for the case of two spatial dimensions in the left panel of Figure A.1. The restriction proceeds as a succession of one–dimensional quadratic interpolations, and is accurate to third order in the grid spacing. Note that the 3-cell-wide fine grid stencil used for this step (nine black circles in the figure) cannot be centered on the parent cell (gray square). In each dimension the stencil includes two fine grid cells on one side of the parent cell and one fine grid cell on the other. The stencil is always positioned so that its center is shifted toward the center of the block (assumed in the figure to be toward the upper left). This ensures that only interior fine grid points, and no fine grid guard cells, are used in this first step.

For the second step, the fine grid guard cells are filled by prolongation from the parent grid. Before the prolongation, the parent grid gets its own guard cells

Thus, third order accuracy for guard cell filling means that the guard cell values have errors of order Δx^3, where Δx is the (fine) grid spacing. Second order accuracy for the evolution code means that, after a finite evolution time, the field variables have errors of order Δx^2.

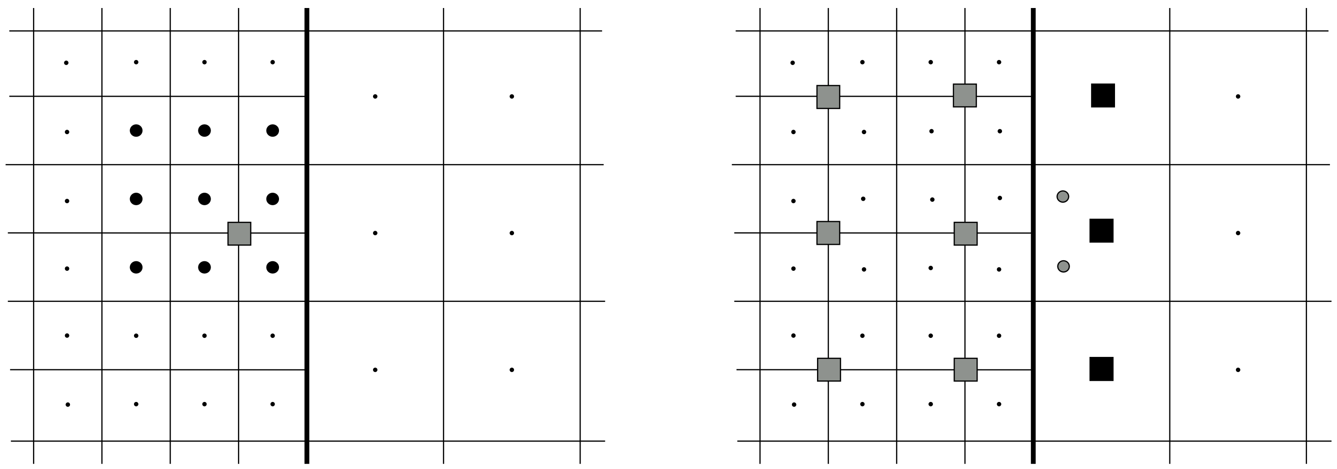


Figure A.1: Guard cell filling in two spatial dimensions. In these pictures, the thick vertical line represents a refinement boundary separating fine and coarse grid regions. The picture on the left shows the first step, in which one of the parent grid cells (gray square) is filled using quadratic interpolation across nine interior fine grid cells (black circles). The other parent grid cells are filled using corresponding stencils of nine interior fine grid cells. (The asymmetry in the left panel is drawn with the assumption that the fine block's center is toward the top-left of the panel.) The picture on the right shows the second step in which two fine grid guard cells (gray circles) are filled using quadratic interpolation across nine parent grid values (squares). These parent grid values include one layer of guard cells (black squares) obtained from the coarse grid region to the right of the interface, and two layers of interior cells (gray squares). The final step in guard cell filling (not shown in this figure) is to use "derivative matching" to fill the guard cells for the coarse grid.

(black squares in the right panel of Figure A.1) from the neighboring grids of the same refinement level, in this case from the coarse grid. The stencil used in the prolongation operation is shown in the right panel of Figure A.1. The prolongation operation proceeds as a succession of one–dimensional quadratic interpolations, and is third order accurate. In this case, the parent grid stencil includes a layer of guard cells (black squares), as well as its own interior grid points (gray squares). At the end of this second step the fine grid guard cells are filled to third order accuracy.

The third step in guard cell filling is "derivative matching" at the interface.[2] With derivative matching the coarse grid guard cell values are computed so that the first derivatives at the interface, as computed on the coarse grid, match the first derivatives at the interface as computed on the fine grid. The first derivative on the coarse grid is obtained from standard second order differencing using a guard cell and its neighbor across the interface. The first derivatives on the fine grid are computed using guard cells and their neighbors across the interface, appropriately averaged to align with the coarse grid cell centers. This third step fills the coarse grid guard cells to third order accuracy.

An alternative to derivative matching, which we do not use, is to fill the coarse grid guard cells from the first layer of interior cells of the parent grid. However, we find that such a scheme leads to unacceptably large reflection and transmission errors for waves passing through the interface. These errors are suppressed by derivative matching.

[2]In Ref. [30] this process is referred to as "flux matching."

Appendix B

Implementation of Misner's Algorithm

In this appendix I delve into more gritty issues regarding my particular implementation of the algorithm for spherical harmonic decomposition that I described and analyzed in Section 4.2. I use, without reintroducing, the notation of that section here. While the details provided here may be quite useful to some readers, those interested only in the theoretical aspects of the work may wish to skip this appendix.

It should be noted that the analysis developed in Section 4.2 was not available when the first version of this code was written. The analysis of the symmetries, for example, was not understood when code development began, and so the approach developed there to treat grids reduced by explicit symmetries was not applied. The approach described in this appendix is a brute force approach. This is less optimal both from the point of view of performance and from the point of view of code development.

My implementation was designed primarily for use with the Hahndol code [30, 55], which itself is layered over the Paramesh AMR libraries [61]. For this reason, the code is written in Fortran 90.[1] While the code was designed primarily for use as part of Hahndol, it is highly modular, and could therefore easily be used with another

[1]I chose Fortran 90 because Paramesh and most of Hahndol are written in that language and I did not wish to deal with C-Fortran interfacing issues. In retrospect this was a mistake because Fortran imposes several limitations that make the code less adaptable than it otherwise would have been. Coding anything that is data structure intensive in Fortran is almost always a mistake. The code is, nonetheless, modular and portable as I describe in the text.

```
call init_misner_harmonics(parfile)
...
call compute_misner_harmonic(l,m,RePhiLM,ImPhiLM,nr,reIndex,imIndex)
```

Figure B.1: The calling sequence for the two "Misner Harmonic" subroutines that form the public interface to the `misner_harmonic` module. The variable `parfile` is a string that is the name of a parameter file. The integer variables `l` and `m` specify which spherical harmonic component to compute on the grid function with real part labeled by index `reIndex` and imaginary part (optional argument) labeled by index `imIndex`. The real and imaginary parts of the spherical harmonic component are returned in `real` variables `RePhiLM` and `ImPhiLM`. The integer variable `nr` indicates where the extraction should take place in a way determined by the parameter file.

code that uses another framework.

B.1 User Interface and Compile-Time Parameters

The module `misner_harmonics` contains the primary portion of the code. Its public interface consists of an initialization routine `init_misner_harmonics` and a computation routine `compute_misner_harmonic`. The argument sequence for these functions is shown in Figure B.1. The first routine initializes the module and must be called after the grid hierarchy is created and before attempting to actually compute a spherical harmonic component. It must be called again every time the grid structure changes, as would happen in a simulation using adaptive mesh refinement. The second function actually computes a specified spherical harmonic component on a specified grid function.

Internally, the initialization subroutine calls a hierarchy of private module functions and subroutines to compute the weights and other required data. This data is stored in module variables, which are enumerated in Figure B.2. The six parameters at the top of the code fragment define compile-time parameters used to set the sizes of the various arrays used in the algorithm.[2] The code comments make these fairly

[2]One would really like these parameters to be run-time choices and the memory allocated dy-

```
module misner_harmonics
  ! Subroutine listing cut for this example
  integer, parameter, private :: NWeightsMax = 11000 ! Max # of weights allowed
  integer, parameter, private :: NRadiiMax   = 5     ! Max # of Radii allowed
  integer, parameter, private :: NMax        = 2     ! Sum over N to this value
  integer, parameter, private :: LMax        = 2     ! Which values of l?
  integer, parameter, private :: SglIndRng   = 9     ! sum of first LMax+1 odds
  integer, parameter, private :: IndRng      = 18    ! SglIndRng*(NMax/2 + 1)
  integer, private :: NWeights(NRadiiMax)            ! # of non-zero weights
  integer, private :: NRadii                         ! # of Radii of extraction
  complex, private :: ThetaLM(SglIndRng,8,NWeightsMax,NRadiiMax)!See note above
  ! (i,j,k,blk) of w_x != 0 in stored in IndexWeights
  integer, private :: IndexWeights(4,NWeightsMax,NRadiiMax)
  logical, private :: initialized = .false.
  logical, private :: symmetry(3) ! Coordinate directions suppressed
  integer, private :: sym_factor  ! 2^(3-N) where N is number of explicit syms
  integer, private :: sym_num     ! 2^N where N is number of explicit symmetries
  real,    private :: sym_sign(8) ! Signs for pseudo-scalars under symmetries
  real,    private :: Delta
  ! Executable code cut for this example
end module misner_harmonics
```

Figure B.2: A code fragment from the `misner_harmonics` module. This shows the internal parameters and data structures used.

self-explanatory. One should note, however, that the parameter `NWeightsMax` limits the number of non-zero weights allowed *on a single processing element.* When the code is run on a parallel machine there are frequently (in fact to use memory optimally there should be) more than `NWeightsMax` total non-zero weights in the full domain. Also note that, since all of the odd n terms in the sum (4.26) vanish identically, only even values of `NMax` are allowed. (There is a run-time check that all of these parameters have valid and compatible values.) In practice one may well want to compute spherical harmonic components at several different radii in a single simulation. The parameter `NRadiiMax` determines the maximum number of extraction radii allowed in a given simulation. One should also note that while `LMax` and `NMax` determine how many values of l are used and the parameter N respectively, they

namically. I hope to include this algorithm improvement in a future revision, likely to be written in C++ for easy of memory management and because a class structure would more naturally allow for the scheme to accommodate several different spin-weighted harmonics in the same simulation without a repetition of code.

cannot independently be increased without limit since the module knows the explicit forms of only the first few spherical harmonics and Legendre polynomials. In order to increase past this limit, higher order harmonics would need to be implemented.[3]

B.2 Internal Data Structures

After the compile-time parameters come four data structures that store most of the mathematical information. The array `NWeights` stores the actual number of non-zero weights computed at each of the extraction radii, while `NRadii` stores the actual number of extraction radii for which weights have been computed.[4] The complex valued array `ThetaLM` contains the quantities

$$\Theta_{lm} = \bar{R}_{lm}(x; R) w_x \tag{B.1}$$

which are exactly the values needed to apply (4.30) to compute a spherical harmonic component. This array has four indices. The first indicates indicates value of l and m (which are mapped to a single integer by equation (B.2) below). The second is used only for grids with explicit symmetries; this is described below. The third is an index labeling the grid points, and corresponds to x in (B.1). The fourth index labels the different extraction radii. Finally, the array `IndexWeights` maps between grid index groups and scalar indices local to the module. Such a mapping is needed because the module only stores non-zero weights. The first index runs over the grid indices (`i`, `j`, `k`, `blk`), which are the index values in the x, y, and z directions augmented by a fourth index used to label the refinement level.[5]

[3]The modules `spherical_harmonics` and `legendre_polynomials` contain this code. These modules will not be discussed further, but are straightforward implementations of the named functions. The functions are combined into several useful combinations in the module `misner_auxiliaries`.

[4]Keep in mind the difference between the parameters which control the *maximum* number of weights and radii allowed in any simulation and the variables which indicated how many weights and radii *are actually used* in a given simulation. The maximums can be changed only by recompiling the code, whereas the actual number used is determined freely (up to the maximum) at run-time.

[5]In a strictly unigrid code the `blk` index would always be set to 1. The name `blk` comes from the basic grid structure in Paramesh, which is called a "block."

The variable `initialized` is used internally to ensure that spherical harmonic components are only computed after the module data has been initialized. Attempting the computation without initializing the module would, of course, be meaningless. The code generates a run-time error in this case.

The variable `Delta` stores the parameter Δ as defined in Section 4.2. It is used only during initialization.

The remaining variables are all related to making the algorithm work when the grid takes advantage of explicit symmetries in the data. This was discussed in detail in Section 4.2.4. Unfortunately the analysis of that section was not available when the original code was written, and so this implementation applies a complicated, brute force approach to the problem. First, at initialization time, any symmetries are recorded in the array `symmetry`. A value of true in the first place, for example, indicates that a reflection symmetry is applied in the x direction. Three true values indicates, then, that only a single octant of data is evolved, with appropriate mirror boundary conditions applied to the data. The variables `sym_factor` and `sym_num` are then defined for later convenience as 2^{3-s} and 2^s, respectively, where s is the number of explicit symmetries. Finally I store, in the first s entries of the array `sym_sign` a value of $+1$ or -1, which indicates the sign acquired by a pseudo-scalar under the corresponding reflection.[6] I will explain the use of these variable in more detail below.

B.3 Internal Functions and Subroutines

There are two stages to the computation, initialization of the data structures, and using the data structures to compute spherical harmonic components. The two subroutines in the public interface, described in Section B.1 correspond to these two tasks. In this section, I describe in some detail what happens internally when these interface subroutines are called by the user. The treatment here is "top down," meaning that I start with the interfaces and descend down through the calling stack.

[6]Entries in the array `sym_sign` beyond position s are not initialized and are never used.

I take this approach because it provides an opportunity to see the big picture before attacking the details.

B.3.1 Initialization

When the user calls `init_misner_harmonics`:

1. A subroutine checks to ensure that the compile-time parameters were given values that are (a) allowed, and (b) self-consistent. If either of these conditions is violated, the code prints an error message and aborts.
2. Reads the parameter file to determine if the grid is reduced by explicit grid symmetries. This means that the array `symmetry` is filled, and the variables, `sym_factor`, `sym_num`, and `sym_sign` are set to appropriate values.
3. Rereads the parameter file to determine how many extraction radii were requested and where those radii are located. These values are stored in local variables; there is a run-time check to ensure that the number of radii requested is less than or equal to the compile-time parameter `NRadiiMax`.
4. Loops over the requested extraction radii and constructs the values stored in the internal data structures. This loop is described below.
5. Sets the variable `initialized` to true.
6. Reports success to standard output and returns control to the main program.

Whenever possible, run-time checks ensure that the values passed into the subroutine by the user (either directly through the calling stack or through the parameter file) are legal values. Illegal values are reported and the code is gracefully ended.

The majority of the work in this process occurs in the loop over extraction radii. The essential portions of the internal routine that runs this loop are shown in Figure B.3. For the most part, each of the subroutines in the loop corresponds to an equation in Ref. [65]. This correspondence is shown in Table B.1. The subroutines

```
subroutine init_misner_harmonics_Rext(Rext, NRext)
  real,    intent(in) :: Rext(:) ! Extraction radii
  integer, intent(in) :: NRext   ! Number of extraction radii
  complex G(IndRng,IndRng), Yadj(IndRng,8,NWeightsMax)
  real    Weight(NWeightsMax), CoordsWeights(3,NWeightsMax)
  integer nr, mype
  do nr = 1, NRext
     NRadii = nr
     call check_radius(Rext(nr),nr,NRext)
     call define_weights(Rext(nr), Weight, CoordsWeights)
     call construct_metric(G, Weight, Rext(nr), CoordsWeights)
     call construct_inverse(G)
     call compute_adjoint_harmonics(Yadj, G, Rext(nr), CoordsWeights)
     call compute_Rlm(Yadj, Rext(nr))
     call store_final_ThetaLM(Weight)
  end do
  initialized = .true.
end subroutine init_misner_harmonics_Rext
```

Figure B.3: This subroutine contains the loop that constructs the values stored in the internal data structures of the `misner_harmonics` module. Variables not declared locally to the subroutine are (private) module variables; cf. Figure B.2. Roughly speaking, each of the subroutines called here corresponds to one equation from the Misner paper.

Subroutine name	**Eqn in Ref. [65]**
`define_weights`	14
`construct_metric`	15
`compute_adjoint_harmonics`	18
`compute_Rlm`	20

Table B.1: The table shows the correspondence between equations in the Misner paper and subroutines in the `misner_harmonics` module.

`check_radius` and `construct_inverse`, which do not correspond directly to equations in the Misner paper, provide a run-time check on the value of the radius and construct the inverse matrix G^{AB} from the matrix G_{AB} using a call to LAPACK [6], respectively. The subroutine `store_final_ThetaLM` also does not correspond to an equation in the Misner paper; it corresponds to (B.1) above. I now descend into the remaining subroutines in some detail.

The `define_weights` subroutine's primary role is to loop over all points in the grid and determine whether or not the volume surrounding them overlaps the extraction shell. In order to do this, however, it must first define the extraction shell, which means defining the parameter Δ. Because I am generically concerned with grids with multiple refinement regions, it is non-trivial to determine the proper value of Δ. In practice I determine it looping over each refinement region and then looping over all points within each refinement region. At each point, I

1. Compute the local grid spacing,
2. Compute the spherical coordinates for the point,
3. Compute a "candidate value of Δ" Δ_c which is some constant (usually 3/4) times the local grid spacing, and
4. Check whether the current point, with the candidate value of Δ_c, would be within the extraction shell. If it would be then I set the actual value of Δ to $\max(\Delta_c, \Delta)$ and move to the next refinement region. If the current point would not be in the extraction shell, then I continue the loop over points in the current refinement region.

At the end of this procedure, Δ is set to the value that it would have if the simulation were unigrid with grid spacing equal to the coarsest refinement level through which the extraction shell could pass. This is, perhaps, clarified by comparing extraction maps (24, 3, 0.25) and (24, 4, 0.5) in Figure 4.3. The shell in (24, 3, 0.25) is "thin" because it is contained entirely within the two finest refinement regions. The shell in (24, 4, 0.5) is "thick" because it *can* reach the outer most region in the map. Note that the thickness of the *shell* in (24, 4, 0.5) is determined by the outer refinement region even though the extraction *sphere* does not reach that far. This is illustrative of what was meant by “could” in the emphasized sentence above.

Once the value of Δ is set, the weights are defined strictly according to (4.28). No information is stored about zero-valued weights. Non-zero weights are stored in a local array, and the indices of the point corresponding to that weight are stored in the array `IndexWeights`. The spherical coordinates of the point are also stored in a local array for later use in the initialization process. The number of weights `NWeights` is incremented for the current extraction radius. Neither the actual value of the weights nor the coordinates of the points are needed once the initialization is complete; neither is it necessary to store them between initialization of different extraction radii. Run-time checks ensure that the number of weights computed does not exceed the compile-time parameter `NWeightsMax`; if the bound is exceeded the code exits gracefully with an error message.

Once the weights are computed, I call the subroutine `compute_metric` to construct the metric G_{AB}. At this point, complications due to explicit grid symmetries become manifest. The metric is defined as an inner product over the entire sphere, whereas only a portion of that sphere will appear in the computational domain when symmetries are enforced explicitly. This was discussed in detail in Section 4.2.4, although that full analysis was not available when I developed this code. Instead I use the procedure outlined in the following pseudo-code to compute the value of G_{AB}:

```
Set G(A,B) to zero
Loop over points with non-zero weight
    Recover the spherical coordinates of the point from local arrays
    Compute $Y_A$ and $Y_B$ at the point
    Add $\bar{Y}_A Y_B w_x$ to the current value of G(A,B)
    Use the array symmetry to check through which planes (if any) to reflect
    For each reflection (or combination of reflections)
        Compute the spherical coordinate of the mirror point (Table 4.2)
        Compute $Y_A$ and $Y_B$ at the mirror point
        Add $\bar{Y}_A Y_B w_x$ to the current value of G(A,B)
    End loop over reflections
End loop over points
In a parallel code, globally reduce to get the sum over all processors
```

The value of the weight w_x is the same at all mirror points.

The metric G_{AB} is inverted with a calls to LAPACK routines [6]. The inverse metric is then used to raise the indices on the spherical harmonics to form the adjoint harmonics. Symmetry issues again arise. Without the analysis of Section 4.2.4 available, I decided to compute the adjoint harmonics separately for each octant when octant symmetry is imposed. (I compute fewer copies when fewer symmetries are used.) The second index of the array `Yadj` in Figure B.3 is just for this purpose. For each multi-index A, I compute Y^A according to the following pseudo-code:

```
Set Yadj to zero
Loop over points with non-zero weight (index i)
    Recover the spherical coordinates of the point from local arrays
    Loop over multi-index $B$
        Compute $Y_B$ at the point
        Add $G^{BA} Y_B$ to Yadj(A,1,i)
        Use the symmetry array to determine the symmetries
        Loop over symmetries (index s starting with s = 2)
            Compute the spherical coordinates of the mirror point (Table 4.2)
            Compute $Y_B$ at the mirror point
            Add $G^{BA} Y_B$ to Yadj(A,s,i)
        End loop over symmetries
    End loop over multi-index $B$
End loop over points
```

In this case G^{AB} is the same for all mirror points; in fact is it the same for all points on the sphere by its definition.

The next step in the process is to compute the functions R^{lm}. The code for this is shown in Figure B.4. To save a bit of memory, I store the values of the functions in the module variables `ThetaLM`. The code simply applies the definition of the functions R^{lm} given by (4.31), which is immediately overwritten in `store_final_ThetaLM` by Θ_{lm} defined in (B.1). This completes the initialization step.

B.3.2 Computation

The ultimate goal of the algorithm, of course, is to compute the spherical harmonic components of grid functions during a simulation. Having initialized the module, the user can accomplish this by calling the subroutine `compute_misner_harmonic`.

```
subroutine compute_Rlm(Yadj, Rext)
  ! Equation 20 from the Misner paper.
  ! Store them in the ThetaLM slots for now.
  use misner_auxilaries
  complex, intent(in) :: Yadj(:,:,:)
  real,    intent(in) :: Rext
  integer n,l,m, A,B
  real    Rn_local
  integer w,s
  ThetaLM(:,1:sym_num,1:NWeights(NRadii),NRadii) = (0.0d0, 0.0d0)
  do w = 1, NWeights(NRadii)
     do n = 0,NMax,2; do l = 0,LMax; do m = -l,l
        A = lm_mapping(l,m)
        B = nlm_mapping(n,l,m)
        Rn_local = Rn(n,Rext,Rext,Delta)
        do s = 1, sym_num
           ThetaLM(A,s,w,NRadii) = ThetaLM(A,s,w,NRadii) &
                + Rn_local*Yadj(B,s,w)
        end do
     end do; end do; end do
  end do
end subroutine compute_Rlm
```

Figure B.4: The subroutine `compute_Rlm`. The module array `ThetaLM` is used as temporary storage, and the function `Rn` computes the function R_n given by (4.22).

This subroutine computes the sum (4.30). The code is shown in Figure B.5. Here the function `i2phi` returns the value of the grid function at the point `i`, `j`, `k`, `blk` when `s` is one. When `s` is not one, it returns the value that the grid function would have at a mirror point assuming the appropriate symmetries. In order to provide the value at mirror points, the routine must make assumptions about whether the grid function is a scalar or pseudo-scalar, and can then use the module array `sym_sign` to determine the required sign.[7]

B.3.3 Index Mappings

I wanted mappings to convert (l, m) pairs and (n, l, m) triples into into single (scalar) indices. This saves space in memory and allowed me to write the code in a way that closely resembles notation used in Ref. [65]. Although making the code resemble

[7]In the text I used this routine to compute the spherical harmonic components of the Weyl scalar Ψ_4, for which the real part behaves as a scalar and the imaginary part behaves as a pseudo-scalar.

```
subroutine compute_misner_harmonic(l, m, RePhiLM, ImPhiLM, nr, &
     phi_rindex, phi_iindex)
  use misner_comm
  integer, intent(in)  :: l,m, nr
  real,    intent(out) :: RePhiLM, ImPhiLM
  integer, intent(in)  :: phi_rindex
  integer, intent(in), optional :: phi_iindex
  complex phiLM
  integer w, s, A, i,j,k,blk
  call compute_misner_harmonic_checks(nr, l, m)
  phiLM = (0.0d0, 0.0d0)
  A = lm_mapping(l,m)
  do w = 1, NWeights(nr)
     i   = IndexWeights(1,w,nr)
     j   = IndexWeights(2,w,nr)
     k   = IndexWeights(3,w,nr)
     blk = IndexWeights(4,w,nr)
     do s = 1, sym_num
        phiLM = phiLM &
             + ThetaLM(A,s,w,nr)*i2phi(i,j,k,blk,s,phi_rindex,phi_iindex)
     end do
  end do
  call mc_communicate_ip(phiLM)
  RePhiLM = real(phiLM)
  ImPhiLM = aimag(phiLM)
end subroutine compute_misner_harmonic
```

Figure B.5: This is the code for the subroutine that computes spherical harmonic components. The function `i2phi` returns the value of the grid function at the point with indices `i`, `j`, `k`, `blk` when `s` is 1, and returns the value of the grid function at points related by appropriate symmetries when `s` is not 1.

the analytic formulas in the Misner paper is not strictly necessary, I consider it good coding practice because it makes the code more readable therefore easier to maintain. I use the mappings

$$\mu(l,m) = l + m + 1 + \sum_{l'=0}^{l}(2l'+1) \tag{B.2}$$

and

$$\nu(n,l,m) = m - l + \sum_{l'=0}^{l}(2l'+1) + \frac{n}{2}\sum_{l'=0}^{l_{\max}}(2l'+1) \tag{B.3}$$

for this purpose. In (B.3), $l_{\max}$ is the maximum value of l allowed by the code and is determined by the compile-time parameter `LMax`. The sum $\Lambda = \sum_{l'=0}^{l_{\max}}(2l'+1)$ is equivalent to the compile-time parameter `SglIndRng`; the parameter was defined just to make the execution of the function implementing (B.3) faster. Defining

$$\mathbf{N}[i,j] = \{k \in \mathbf{N} \mid i \leq k \leq j\} \tag{B.4a}$$

$$S_1 = \{(l,m) \mid l \in \mathbf{N}[0, l_{\max}], m \in \mathbf{N}[-l, l]\} \tag{B.4b}$$

$$S_2 = \{(n,l,m) \mid n \in \mathbf{N}[0,N] \text{ is even}, (l,m) \in S_1\} \tag{B.4c}$$

it is easy to check that μ is a bijection from S_1 onto $\mathbf{N}[1,\Lambda]$ and ν is a bijection from S_2 onto $\mathbf{N}[1, N\Lambda/2]$. This is exactly what is needed in order to have a unique and memory efficient mapping of index groups into one dimensional indices.

In the code examples above, the function `lm_mapping` implements μ, and the function `nlm_mapping` implements ν.

B.4 Notes on Modularity

I noted above that this code is quite modular. It could, in fact, be used with a code other than Hahndol with minimal rewriting. To make the module work with another code, the following pieces would need to be rewritten:

1. The parameter file reader must be rewritten to match the parameter syntax of the new host code.

2. The functions in Table B.2 must be implemented to provide basic grid information to the module.

3. The macros `ht_traverse` and `ht_traversal` (described below) must be provided.

4. The helper module `misner_comm` would need to be rewritten only if a communication scheme other than MPI is used.

The subroutine `ht_mype`, one of the functions listed in Table B.2, should return 0 for the head node in a parallel code. It does not, in fact, matter what it returns for other nodes, although in an MPI based code it would most naturally return the processing element's MPI index. The fourth index `blk` used by several of the routines

Subroutine name	**Arguments**	**Returns**	**Module**
`ht_mype`	None	Integer	`ht_comm`
`hd_comp_grid_spacing`	`blk`, `dx`	-	`hd_coordinates`
`hd_comp_coordinates`	`blk`, `i`, `j`, `k`, `x`	-	`hd_coordinates`
`unk2scalar`	`indx`, `blk`, `i`, `j`, `k`, `phi`	-	`unk2local`

Table B.2: The table shows which subroutines and functions must be provided in order for the module to work with a code other than Hahndol. The first function, `ht_mype` returns the processor number for parallel codes. The variables `blk`, `i`, `j`, and `k` are integers and inputs to the subroutines listed. The variables `dx` and `x` are three dimensional arrays of real numbers output by the subroutines, and `phi` is real variable output by the subroutine.

in the table is used to distinguish between different refinement regions. A unigrid code should always pass 1 in that position; an FMR/AMR code is free to use it, in coordination with the macros below, to sensibly deal with its refinement scheme.

The macro pair `ht_traverse` and `ht_traversal` must be provided to loop over the grid in a sensible way. For unigrid definitions similar to

```
#define ht_traverse(I,J,K,BLK) (.true.) then; \\
        do K = 1, NPTS; do J = 1, NPTS; do I = 1, NPTS
#define ht_traversal do; end do; end do; end if
```

should work. They are called in the code with the syntax

```
if ht_traverse(i,j,k,blk)
   ! Executable statements here
end ht_traversal
```

which explains the seemingly strange definitions. In Hahndol we do not want to loop over all blocks for technical reasons specific to the data structures used by Paramesh to organize the FMR. This is the reason for the if-statement. Other FMR schemes may or may not find it useful, but for compatibility it must remain even if the test in some other code is trivial (as it is in the sample definition provided here).

There are other modules used by `misner_harmonics`, but they are all completely self-contained. The only exception could be `misner_comm`, which is currently written to support the MPI protocol only. A code using another message passing scheme (or none at all), would have to rewrite this module as well. This would most likely be trivial as it only contains two subroutines, and both are simple.

Bibliography

[1] Miguel Alcubierre, *Hyperbolic slicings of spacetime: singularity avoidance and guage shocks*, Phys. Rev. D (2003), 607.

[2] Miguel Alcubierre, Gabrielle Allen, and Gerd Lanfermann, *Cactus 4.0 Thorn Guide (Stable Release)*, See the chapter on the `Boundary` thorn in the documentation at `http://www.cactuscode.org`, January 2003.

[3] Miguel Alcubierre and Bernd Brügmann, *Simple excision of a black hole in 3+1 numerical relativity*, Phys. Rev. D **63** (2001), 104006.

[4] Miguel Alcubierre, Bernd Brügmann, Peter Diener, Michael Koppitz, Denis Pollney, Edward Seidel, and Ryoji Takahashi, *Gauge conditions for long-term numerical black hole evolutions without excision*, Phys. Rev. D **67** (2003), 084023.

[5] Miguel Alcubierre, Bernd Brügmann, Thomas Dramlitsch, José A. Font, Philippos Papadopoulos, Edward Seidel, Nikolaos Stergioulas, and Ryoji Takahashi, *Towards a stable numerical evolution of strongly gravitating systems in general relativity: The conformal treatments*, Phys. Rev. D **62** (2000), 044034.

[6] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, *LAPACK users' guide*, third ed., Society for Industrial and Applied Mathematics, Philadelphia, 1999.

[7] Matthew Anderson and Richard Matzner, *Extended lifetime in computational evolution of isolated black holes*, gr-qc/0307055 (2003).

[8] Peter Anninos, Karen Camarda, Joan Massó, Edward Seidel, Wai-Mo Suen, and John Towns, *Three dimensional numerical relativity: the evolution of black holes*, Phys. Rev. D **52** (1995), 2059–2082.

[9] R. Arnowitt, S. Deser, and C. W. Misner, *The dynamics of general relativity*, Gravitation: An Introduction to Current Research (L. Witten, ed.), Wiley, New York, 1962, pp. 227–265.

[10] John Baker, Manuella Campanelli, and Carlos O. Lousto, *The Lazarus project: A pragmatic approach to binary black hole evolutions*, Phys. Rev. D **65** (2002), 044001.

[11] Barry Barish, *First generation interferometers*, in Centrella [27], p. 3.

[12] Thomas W. Baumgarte and Stuart L. Shapiro, *Numerical integration of Einstein's field equations*, Phys. Rev. D **59** (1999), 024007.

[13] Z. B. Belanger, *Adaptive mesh refinement in the T 2 symmetric spacetime*, Master's thesis, Oakland University, 2001.

[14] P. Bender, A. Briller, I. Ciufolini, A. M. Cruise, C. Cutler, K. Danzmann, F. Fidecaro, W. M. Folkner, J. Hough, P. McNamara, M. Peterseim, D. Robertson, M. Rodrigues, A. Rüdiger, M. Sandford, G. Schäfer, R. Schilling, B. Schutz, C. Speake, R. T. Stebbins, T. Sumner, P. Touboul, J.-Y. Vinet, S. Vitale, H. Ward, and W. Winkler, *LISA, pre-phase A report*, On-line resource [cited July 11, 2004] `http://www.srl.caltech.edu/lisa/documents`, 1998.

[15] M. Berger and J. Oliger, *Adaptive mesh refinement for hyperbolic partial differential equations*, J. Comp. Phys. **53** (1984), 484.

[16] D. Bernstein, *A numerical study of the black hole plus brill wave spacetime*, Ph.D. thesis, University of Illinois at Urbana-Champaign, 1993.

[17] C. Bona, J. Massó, E. Seidel, and J. Stela, *New formalism for numerical relativity*, Phys. Rev. Lett. **75** (1995), 600.

[18] J. Bowen and J. W. York, *Time-asymmetric initial data for black holes and black-hole collisions*, Phys. Rev. D **21** (1980), 2047.

[19] D. Brill and R. Lindquist, *Interaction energy in geometrostatics*, Phys. Rev. **131** (1963), 471–476.

[20] Dieter S. Brill, *On the positive definite mass of the Bondi-Webber-Wheeler time-symmetric gravitational waves*, Ann. Phys. **7** (1959), 466.

[21] Othmar Brodbeck, Simonetta Frittelli, Peter Hübner, and Oscar A. Reula, *Einstein's equations with asymptotically stable constraint propagation*, J. Math. Phys. **40** (1999), no. 2, 909.

[22] David Brown and Lisa Lowe, *AMRMG (Adaptive Mesh Refinement MultGrid Code)*, Penn State Numerical Relativity Lunch Talk, February 2003.

[23] Bernd Brügmann, *Adaptive mesh and geodesically sliced Schwarzschild spacetime in 3+1 dimensions*, Phys. Rev. D **54** (1996), 7361–7372.

[24] ———, *Binary black hole mergers in 3D numerical relativity*, Int. J. Mod. Phys. **8** (1999), 85.

[25] Bernd Brügmann, Wofgang Tichy, and Nina Jansen, *Numerical simulation of orbiting black holes*, gr-qc/0312112 (2003).

[26] Gioel Calabrese, *A remedy for constraint growth in numerical relativity*, gr-qc/0404036 (2004).

[27] Joan M. Centrella (ed.), *Astrophysical sources for ground-based gravitational wave detectors*, American Institute of Physics, Melville, New York, 2001.

[28] S. Chandrasekhar, *The mathematical theory of black holes*, Oxford University Press, 1992.

[29] G. Chesshire and W. D. Henshaw, *Composite overlapping meshes for the solution of partial-differential equations*, J. Comp. Phys. **90** (1990), 1.

[30] Dae-Il Choi, J. David Brown, Breno Imbiriba, Joan Centrella, and Peter MacNeice, *Interface conditions for wave propagation through mesh refinement boundaries*, J. Comp. Phys. **193** (2004), 398–425.

[31] M. W. Choptuik, *Experiences with an adaptive mesh refinement algorithm in numerical relativity*, in Evans et al. [40].

[32] ______, *Universality and scaling in gravitational collapse of massless scalar field*, Phys. Rev. Lett. **70** (1993), 9.

[33] Matthew W. Choptuik, Eric W. Hirschmann, Steven L. Liebling, and Frans Pretorius, *Critical collapse of the massless scalar field in axisymmetry*, Phys. Rev. D **68** (2003), 044007.

[34] G. B. Cook, M. F. Huq, S. A. Klasky, M. A. Scheel, A. M. Abrahams, A. Anderson, P. Anninos, T. W. Baumgarte, N. T. Bishop, S. R. Brandt, J. C. Browne, K. Camarda, M. W. Choptuik, R. R. Correll, C. R. Evans, L. S. Finn, G. C. Fox, R. Gómez, T. Haupt, L. E. Kidder, P Laguna, W. Landry, L. Lehner, J. Lenaghan, R. L. Marsa, J. Masso, R. A. Matzner, S. Mitra, P. Papadopoulos, M. Parashar, L. Rezzolla, M. E. Rupright, F. Saied, P. E. Saylor, E. Seidel, S. L. Shapiro, D. Shoemaker, L. Smarr, W. M. Suen, B. Szilágyi, S. A. Teukolsky, M. H. P. M. van Putten, P. Walker, J. Winicour, and J. W. York, Jr., *Boosted three-dimensional black-hole evolutions with singularity excision*, Phys. Rev. Lett. **80** (1998), 2512.

[35] T. Damour and J. H. Taylor, *On the orbital period change of the binary pulsar PSR 1913+16*, ApJ **366** (1991), 501.

[36] ______, *Strong-field test of relativistic gravity and binary pulsars*, Phys. Rev. D **45** (1992), 1840.

[37] Steven Detweiler, *Evolution of the constraint equations in general relativity*, Phys. Rev. D **35** (1987), 1095.

[38] Albert Einstein, *Der Feldgleichungen der Gravitation*, Sitzungsberiche der Deutschen Akademie der Wissenschaften zu Berlin, Klasse fur Mathematik, Physik, und Technik (1915), 844.

[39] K. Eppley, *The numerical evolution of the collision of two black holes*, Ph.D. thesis, Princeton University, Princeton, New Jersey, 1975.

[40] C. Evans, L. Finn, and D. Hobill (eds.), *Frontiers in numerical relativitiy*, Cambridge University Press, Cambridge, 1989.

[41] C. R. Evans, *Enforcing the momentum constraints during axisymmetric spacelike simulations*, in Evans et al. [40].

[42] David R. Fiske, *Toward making the constraint hypersurface an attractor in free evolution*, Phys. Rev. D **69** (2004), 047501.

[43] P. Fritschel, *The second generation LIGO interferometers*, in Centrella [27], p. 15.

[44] Simonetta Frittelli, *Note on the propagation of the constraints in standard 3+1 general relativity*, Phys. Rev. D **55** (1997), 5992.

[45] J. N. Goldberg, A. J. MacFarlane, E. T. Newman, F. Rohrlich, and E. C. G. Sudarshan, *Spin-s spherical harmonics and* ð, J. Math. Phys. **8** (1967), 2155.

[46] Herbert Goldstein, *Classical mechanics*, second ed., Addison-Wesley, 1980.

[47] S. G. Hahn and R. W. Lindquist, *The two-body problem in geometrodynamics*, Ann. Phys. **29** (1964), 304.

[48] William O. Hamilton, *Resonant detectors of gravitational radiation*, in Centrella [27], p. 24.

[49] W. D. Henshaw and D. W. Schwendeman, *An adaptive numerical scheme for high-speed reactive flow on overlapping grids*, J. Comp. Phys. **191** (2003), 420.

[50] Simon David Hern, *Numerical relativity and inhomogeneous cosmologies*, Ph.D. thesis, Cambridge University, 1999.

[51] R. A. Hulse, *The discovery of the binary pulsar*, Les Prix Nobel, The Nobel Foundation, 1994.

[52] R. A. Hulse and J. H. Taylor, *A high sensativity pulsar survey*, ApJ **191** (1974), L59.

[53] ———, *A deep sample of new pulsars and their spatial extent in the galaxy*, ApJ **201** (1975), L55.

[54] ———, *Discovery of a pulsar in a binary system*, ApJ **195** (1975), L51.

[55] Breno Imbiriba, John Baker, Dae-Il Choi, Joan Centrella, David R. Fiske, J. David Brown, James van Meter, and Kevin Olson, *Evolving a puncture black hole with fixed mesh refinement*, gr-qc/0403048 (2004).

[56] Lawrence E. Kidder, Mark A. Scheel, and Saul A. Teukolsky, *Extending the lifetime of 3D black hole computations with a new hyperbolic system of evolution equations*, Phys. Rev. D **64** (2001), 064017.

[57] W. Kinnersley, *Type D vacuum metrics*, J. Math. Phys. **10** (1969), 1195.

[58] A. M. Knapp, E. J. Walker, and T. W. Baumgarte, *Illustrating stability properties of numerical relativity in electrodynamics*, Phys. Rev. D **65** (2002), 064031.

[59] Pablo Laguna and Deirdre Shoemaker, *Numerical stability of a new conformal-traceless 3 + 1 formulation of the Einstein equation*, Class. Quant. Grav. **19** (2002), 3679.

[60] Lee Lindblom, Mark A. Scheel, Lawrence E. Kidder, Harald P. Pfeiffer, Deirdre Shoemaker, and Saul A. Teukolsky, *Controlling the growth of the constraints in hyperbolic evolution systems*, gr-qc/0402027 (2004).

[61] P. MacNeice, K. Olson, C. Mobarry, R. de Fainchtein, and C. Packer, *PARAMESH: a parallel adaptive mesh refinement community toolkit*, Comput. Phys. Comm. **126** (2000), 330.

[62] Maplesoft, *Maple*, 615 Kumpf Drive, Waterloo, Ontario Canada N2V 1K8.

[63] Richard A. Matzner, H. E. Seidel, Stuart L. Shapiro, L. Smarr, W.-M. Suen, Saul A. Teukolsky, and J. Winicour, *Geometry of a black hole collision*, Science **270** (1995), 941–947.

[64] Charles W. Misner, *The method of images in geometrostatics*, Ann. Phys. **24** (1963), 102.

[65] ______, *Spherical harmonic decomposition on a cubic grid*, Class. Quant. Grav. **21** (2004), S243.

[66] Charles W. Misner, Kip S. Thorne, and John Archibald Wheeler, *Gravitation*, W. H. Freeman and Company, New York, 1970.

[67] Gabriel Nagy, Omar E. Ortiz, and Oscar A. Reula, *Strongly hyperbolic second order Einstein's evolution equations*, gr-qc/04022123 (2004).

[68] Kimberly C. B. New, Dae-Il Choi, Joan M. Centrella, Peter MacNeice, Mijan Huq, and Kevin Olson, *Three-dimensional adaptive evolution of gravitational waves in numerical relativity*, Phys. Rev. D **62** (2000), 084039.

[69] E. T. Newman and R. Penrose, *Note on the Bondi-Metzner-Sachs group*, J. Math. Phys. **7** (1966), 863.

[70] P. Papadopoulos, E. Seidel, and L. Wild, *Adaptive computation of gravitational waves from black hole interactions*, Phys. Rev. D **58** (1998), 084002.

[71] Michael E. Peskin and Daniel V. Schroeder, *An introduction to quantum field theory*, Perseus Books, Reading, Massachusetts, 1995.

[72] Richard H. Price, *Nonspherical perturbations of relativistic gravitational collapse. II. Integer-spin, zero-rest-mass fields*, Phys. Rev. D **5** (1972), 2439.

[73] Erik Schnetter, *Gauge fixing for the simulation of black hole spacetimes*, Ph.D. thesis, Eberhard-Karls-Universität zu Tübingen, 2003.

[74] Erik Schnetter, Scott H. Hawley, and Ian Hawke, *Evolutions in 3D numerical relativity using fixed mesh refinement*, Class. Quant. Grav. **21** (2004), 1465–1488.

[75] Bernard F. Schutz, *Gravitational wave sources: An overview*, The Astrophysics of Gravitational Wave Sources (Joan M. Centrella, ed.), American Institute of Physics, Melville, New York, 2003, pp. 3–26.

[76] Karl Schwarzschild, *Über das Gravitationsfeld eines Massenpunktes nach der Einsteinschen Theorie*, Sitzungsberiche der Deutschen Akademie der Wissenschaften zu Berlin, Klasse fur Mathematik, Physik, und Technik (1916), 189–196.

[77] Masaru Shibata and Takashi Nakamura, *Evolution of three-dimensional gravitational waves: Harmonic slicing case*, Phys. Rev. D **52** (1995), 5428.

[78] Hisa-aki Shinkai and Gen Yoneda, *Adjusted ADM systems and their expected stability properties: constraint propagation analysis in Schwarzschild spacetime*, Class. Quant. Grav. **19** (2002), 1027.

[79] Deirdre Shoemaker, Kenneth Smith, Ulrich Sperhake, Pablo Laguna, Erik Schnetter, and David Fiske, *Moving black holes via singularity excision*, Class. Quant. Grav. **20** (2003), 3729–3744.

[80] Larry Smarr, *The structure of general relativity with a numerical example*, Ph.D. thesis, University of Texas, Austin, Austin, Texas, 1975.

[81] ______, *Spacetimes generated by computers: Black holes with gravitational radiation*, N. Y. Acad. Sci **302** (1977), 569.

[82] I. H. Stairs, Z. Arzoumanian, F. Camilo, A. G. Lyne, D. J. Nice, and J. H. Taylor, *Measurement of relativistic orbital decay in the PSR B1534+12 binary system*, ApJ **505** (1998), 352.

[83] I. H. Stairs, S. E. Thorsett, J. H. Taylor, and A. Wolszczan, *Studies of the relativistic binary pulsar PSR B1534+21. I. Timing analysis*, ApJ **581** (2002), 501.

[84] J. H. Taylor, *Binary pulsars and relativistic gravity*, Rev. Mod. Phys. **66** (1994), 711.

[85] Saul A. Teukolsky, *Pertubations of a rotating black hole. I. Fundemental equations for gravitational, electromagnetic, and neutrino-field perturbations*, ApJ **185** (1973), 635.

[86] ______, *Linearized quadrupole waves in general relativity and the motion of test particles*, Phys. Rev. D **26** (1982), 745.

[87] ______, *On the stability of the iterated Crank-Nicholson method in numerical relativity*, Phys. Rev. D **61** (2000), 087501.

[88] Manuel Tiglio, *Dynamical control of the constraints growth in free evolutions of Einstein's equations*, gr-qc/0304062 (2003).

[89] Robert M. Wald, *General relativity*, University of Chicago Press, 1984.

[90] Karen Willacy, *LISA newsletter*, Available online [cited July 11, 2004] from `http://lisa.jpl.nasa.gov/news.html`, April 2004.

[91] J. Winicour, *Characteristic evolution and matching*, Living Rev. Relativity **4** (2001), 3, [Online article]: cited July 13, 2004, `http://www.livingreviews.org/lrr-2001-3`.

[92] A. Wolszczan, *A nearby 37.9 ms radio pulsar in a relativistic binary system*, Nature **350** (1991), 688.

[93] Hwei-Jang Yo, Thomas W. Baumgarte, and Stuart L. Shapiro, *Improved numerical stability of stationary black hole evolution calculations*, Phys. Rev. D **66** (2002), 084026.

[94] Gen Yoneda and Hisa-aki Shinkai, *Constraint propagation in the family of ADM systems*, Phys. Rev. D **63** (2001), 124019.

[95] ______, *Advantages of a modified ADM formulation: Constraint propagation analysis of the Baumgarte-Shapiro-Shibata-Nakamura system*, Phys. Rev. D **66** (2002), 124003.

[96] James W. York, Jr., *Kinematics and dynamics of general relativity*, Sources of Gravitational Radiation (Larry Smarr, ed.), Cambridge University Press, Cambridge, England, 1979, pp. 83–126.

Vita

David Fiske was born on January 12, 1977 in Troy, New York to Donna and Roger Fiske. He completed his primary education at St. Michael School, Donnell Junior High School, and Findlay Senior High School, all in Findlay, Ohio. As part of his high school coursework, he took physics classes at the University of Findlay. He was the Findlay High School salutatorian in 1995.

After high school, David attended the Ohio State University in Columbus, Ohio from 1995 to 1999. While attending Ohio State he spent nearly two years working with the Relativistic Heavy Ion Collision group under the supervision of Prof. Thomas Humanic and Dr. Ivan Kotov. His work developing equipment to test the quality of silicon drift detectors for use in the RHIC STAR experiment lead to his senior honors thesis and a journal publication. David was also a Laboratory Instructor, supervising and grading the work of students in undergraduate physics lab courses. He received a Bachelor of Science degree with Distinction in Physics and a Bachelor of Science degree with Distinction in Mathematics, while graduating Summa Cum Laude, and with Honors in the Liberal Arts.

During summers David also participated in National Science Foundation Research Experience for Undergraduates Programs at the Center for the Neural Basis of Cognition, a joint venture of Carnegie Mellon University and the University of Pittsburgh (1997); and at the Department of Physics and Astronomy at the University of California, Irvine (1998). At the CNBC, he attempted to construct a neuronal network to understand certain behaviors observed in honey bees. At UCI, he worked in Prof. Riley Newman's gravity lab on cryogenic torsion experiments designed to measure Newton's gravitational constant and to test the equivalence principle.

As a graduate student at the University of Maryland, College Park, beginning in 1999, David worked for one year with Prof. Alex Dragt studying applications of Lie algebras to particle accelerator design, before beginning his present work with Prof. Charles Misner in numerical relativity. During his time with Prof. Misner, David also spent time as a visitor at the Pennsylvania State University under the supervision of Prof. Pablo Laguna, and at NASA Goddard Space Flight Center under the supervision of Dr. Joan Centrella. David also taught discussion sections for undergraduate physics courses. He has authored or co-authored four journal publications during his time at the University of Maryland (all of which appear in the Bibliography). David received his Masters of Science degree in Physics from the University of Maryland in 2003, and, upon completion of his doctoral degree, will become a National Research Council Post-doctoral Fellow at NASA Goddard Space Flight Center.

David lives in College Park, Maryland, and is engaged to marry long-time friend and fellow graduate student Violeta Prieto on October 16, 2004 on the campus of the University of Maryland, College Park.

Publications

- Breno Imbiriba, John Baker, Dae-Il Choi, Joan Centrella, David R. Fiske, J. David Brown, James van Meter, and Kevin Olson. "Evolving a puncture black hole with fixed mesh refinement," (submited to *Phys. Rev. D*), gr-qc/0403048.

- Miguel Alcubierre, Gabrielle Allen, Carles Bona, David Fiske, Tom Goodale, F. Siddartha Guzmán, Ian Hawke, Scott H. Hawley, Sascha Husa, Michael Koppitz, Christiane Lechner, Denis Pollney, David Rideout, Marcelo Salgado, Erik Schnetter, Edward Seidel, Hisa-aki Shinkai, Deirdre Shoemaker, Béla Szilágyi, Ryoji Takahashi, and Jeff Winicour. "Toward standard testbeds for numerical relativity," *Class. Quant. Grav.* **21** (2004), 589.

- David R. Fiske. "Toward making the constraint hypersurface an attractor in free evolution," *Phys. Rev. D* **69** (2004), 047501.

- Deirdre Shoemaker, Kenneth Smith, Ulrich Sperhake, Pablo Laguna, Erik Schnetter, and David Fiske. "Moving black holes via singularity excision," *Class. Quant. Grav.* **20** (2003), 3729.

- A. Asumus, R. Bellwied, R. Beuttenmuller, W. Chen, H. Dyke, V. Eremin, D.R. Fiske, G.W. Hoffmann, T.J. Humanic, M. Grau, I. Ilyashenko, I.V. Kotov, H.W. Kraner, B. Leonhardt, Z. Li, D. Lynn, S.U. Pandey, J. Schambach, J. Sedlmeir, E. Sugarbaker, and J. Takahashi. "Probe station testing of large area silicon drift detectors," *IEEE Trans. Nucl. Sci.* **47** (2000), 1375.