universität
wien

MASTERARBEIT

Titel der Masterarbeit

Hybrid numerical methods for black hole puncture evolution

Verfasser

Mag.rer.nat. Kurt Fritz BSc

angestrebter akademischer Grad

Master of Science (MSc)

Wien, 2013

Studienkennzahl lt. Studienblatt: A 066 876
Studienrichtung lt. Studienblatt: Masterstudium Physik UG2002
Betreuer: Ao. Univ.-Prof. Dr. Robert Beig

Contents

0 Introduction

Through the last decades the applicability of numerical methods to problems in general relativity has vastly increased due both to the exceptional growth of computational power and new simulation techniques. These problems include cosmological models, galaxies, stars, black holes and gravitational waves.

The well known ADM formulation of general relativity turned out to be unstable numerically as it is only weakly hyperbolic in general. In order to solve these problems, new formulations of Einsteins equations with better stability properties were developed, which made stable long time numerical simulations possible.

Among these formulations are the generalized harmonic formulation [18, 25] and the Baumgarte – Shapiro – Shibata – Nakamura – Oohara – Kojima formulation, commonly referred to as the BSSN(OK) system [4, 27, 23] and described in Chapter 2. It became famous among other applications through the simulation of black hole binary coalescences and the accurate computation of the emitted gravitational waves (see also [24, 13, 2, 11, 12]).

BSSN simulations customarily use the so called *finite difference method* described in Section 1.2. Although this technique leads to satisfactory results, there exist more sophisticated methods which have been successfully applied to the generalized harmonic formulation of the Einstein equations and lead to a reduction of computational costs for a desired accuracy.

One of these methods is the *pseudospectral method*, which was also initially used with the Spectral Einstein Code (SpEC) [21] for simulations in the generalized harmonic formulation of general relativity and is described in Section 1.3. Instead of local Taylor polynomials, on which the finite difference methods rely, this global approach uses polynomial bases for functions on whole grid patches with gridpoints distributed in a nonuniform way. Note that SpEC uses more sophisticated spectral methods with improved stability properties by including penalty terms [28]. The physical domain is usually decomposed into many patches.

So far spectral methods have not led to satisfying results for the evolution of black holes using puncture methods and the BSSN system. The idea of this thesis is that instead of replacing the numerical method, we can combine the two approaches and benefit from the advantages of both.

It is usually a good idea to test a new numerical method for solving Einstein's equations in spherical symmetry, which reduces the amount of variables from 16 to 9 (and was suggested in various papers on this topic, see also [16] and [8]). An additional advantage of restricting ourselves to spherical symmetry lies in the computability of the problems on a workstation. The price of this simplification is that we can only evolve a single black hole and there is no

gravitational radiation.

This thesis concentrates on the crucial point of the combination of the two numerical methods, namely the formulation of the *characteristic* interface between a finite difference grid segment near to the origin and a pseudospectral grid segment covering the outer region. The actual implementation requires an in-depth knowledge of the BSSN system and care in choosing the parameters.

We give an overview of the required numerical methods in Chapter 1. This is followed by the derivation of the generalized BSSN system in Chapter 2, which allows a better treatment of spherical symmetry than the usual BSSN system.

In Chapter 3 some necessary properties of partial differential equations are introduced along with toy models for the evaluation of the simulation techniques and their implementation. Additionally we apply a characteristic analysis to the generalized BSSN system in spherical symmetry which leads to characteristic variables required in the interface formulation.

Chapter 4 and Chapter 5 deal with the wave equation in one dimension and in three dimensions with spherical symmetry. Here we check the implementation of the known methods and we gain experience for the formulation of the interface conditions. Moreover we evaluate the boundary conditions which are used in a similar manner for the BSSN system later on.

Finally, Chapter 6 and Chapter 7 deal with the actual implementation of the BSSN system and the simulation results. While the first one treats initial and boundary conditions and the elementary implementation, the latter one focuses on the long time behavior of the system. Filtering techniques are applied and the configuration of the problem is chosen in a way which enhances long time stability.

At this point I want to thank my advisor Dr. Michael Pürrer for the idea to this thesis and his constant support to tackle all arising problems. In spite of the far distance to Cardiff, he was always easy to address with upcoming questions or fast to suggest an alternative, if something did not work out as expected.

Furthermore I want to thank Prof. Robert Beig for taking the role as my administrative supervisor for the University of Vienna. Additional thanks to the faculty of physics and in particular to the gravitation group for the allowance of using the workstation Pauli and the possibility of presenting my thesis in the 3rd Central European Relativity Seminar in Golm 2013.

1 Numerical methods

This chapter is devoted to the numerical methods used in this thesis. We discuss how a physical domain can be represented on a numerical grid, introduce differentiation methods and explain the concept of numerical convergence and filtering.

1.1 Discretization of the domain

As a reference for the discussion in this section we use [5, p. 188ff]. After the choice of a coordinate system on the spacetime manifold, or on one or more patches that cover it, we can consider numerical approximations of functions $f(t, x)$ on this manifold. In the rest of this work we assume spherical symmetry (as defined in Section 2.3) and thus only one time and one space coordinate are needed. The previously mentioned approximations are represented by values on a discrete set of points, a so called numerical *grid*. In the following we define special properties of a grid for the one dimensional case. For more dimensions appropriate generalizations have to be considered.

1.1.1 Vertex-centered and cell-centered grids

First we can divide the interval $[x_{\min}, x_{\max}] \subseteq \mathbb{R}$ into N *grid cells*.

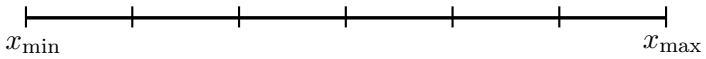


Figure 1.1: The interval $[x_{\min}, x_{\max}]$ is divided into $N = 6$ grid cells.

We can choose our grid points to be located either at the center of these cells, which is referred to as a *cell-centered grid*,

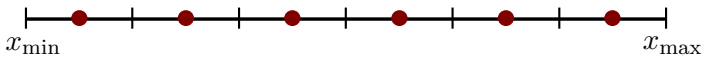


Figure 1.2: Cell-centered grid with $N = 6$ grid cells and 6 grid points.

or on the vertices, which is refered to as a *vertex-centered grid*.

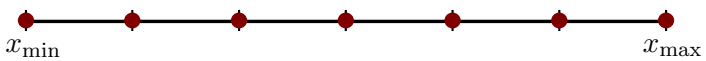


Figure 1.3: Vertex-centered grid with $N = 6$ grid cells and 7 grid points.

The distance between the grid points x_i is called *grid spacing* Δx and may in principle depend on x (or rather x_i). For *uniform grids*, which means that Δx is constant, we get

$$x_i = x_0 + i \cdot \Delta x. \tag{1.1}$$

1.1.2 Time discretization

If the solution depends on time we also discretize the time coordinate, for example:

$$t^n = t^0 + n \cdot \Delta t \tag{1.2}$$

where the superscript n denotes the n-th time level and should not be confused with an exponent.

1.1.3 The discretized spacetime

In the examples of this thesis we consider only a vertex centered discretization of the time coordinate with a uniform grid spacing Δt which we will refer to as the *time step size* or *time step*. The spatial grid at a given time t^n will be called the *time slice at* t^n or shorter *time slice*.

1.2 Finite difference method

Following [5, p. 192] we describe two well suited grids for the implementation of finite difference methods. The first one is a cell-centered uniform grid with N grid points located at

$$x_i = x_{\min} + \left(i - \frac{1}{2}\right) \Delta x \qquad i = 1, \ldots, N. \tag{1.3}$$

The second one is a vertex-centered uniform grid with $N + 1$ grid points located at

$$x_i = x_{\min} + (i - 1) \, \Delta x \qquad i = 1, \ldots, N + 1. \tag{1.4}$$

The difference between cell-centered and vertex-centered grids only affects the implementation of boundary conditions, but not the finite difference representation of the differential equation itself. So we will first calculate the formulas required in the simulations and turn back to the question of implementing boundary conditions when we consider the implementation of the examples.

1.2.1 Fundamentals of finite difference methods

We assume that the spacetime is already discretized as discussed above using a uniform grid. By [5, p. 189] the *finite difference representation* of a function $f(t, x)$ on spacetime is given

by

$$f_i^n = f(t^n, x_i) + \text{truncation error}. \tag{1.5}$$

The *truncation error* results from "truncating" the Taylor expansion of a function. Due to the finite numerical precision available for floating point operations on CPUs, we stop our approximation at a certain order in the gridspacing. So f_i^n only approaches the correct value of f at t^n and x_i as the finite difference solution converges to the correct solution (the truncation error goes to zero).

The discretization of a differential equation requires also expressions for derivatives. Consider for the moment a function $f(x)$ which can be differentiated to sufficiently high order. Then its Taylor series is given by

$$f_{i+1} = f(x_i + \Delta x) = f(x_i) + \Delta x \cdot (\partial_x f)|_{x_i} + \frac{(\Delta x^2)}{2} \cdot (\partial_x^2 f)|_{x_i} + \mathrm{O}\left(\Delta x^3\right) \tag{1.6}$$

Solving for $(\partial_x f)|_{x_i} = (\partial_x f)_i$ we find

$$(\partial_x f)_i = \frac{f_{i+1} - f_i}{\Delta x} + \mathrm{O}\left(\Delta x\right). \tag{1.7}$$

In the limit $\Delta x \to 0$ this is just the definition of the partial derivative. The truncation error of this expression is linear in Δx.

Subtracting the analogous Taylor expansion to the point x_{i-1} from Equation 1.6 and solving again for $(\partial_x f)_i$ gives

$$(\partial_x f)_i = \frac{f_{i+1} - f_{i-1}}{2\Delta x} + \mathrm{O}\left(\Delta x^2\right). \tag{1.8}$$

This expression is now second order in Δx, i.e. the truncation error drops by a factor of four when we reduce the grid spacing by a factor of two. The key idea is, that after the combination of the two Taylor approximations, the leading order error terms cancel out and we get a higher order representation of the derivative. This works only for uniform grids.

As this gets more complicated with higher order finite difference methods we will follow a more direct way for the calculation of the weights of the function values in the derivative formulas. Most of the resulting weights are also given in [17], where only the centered-left- and centered-right-sided values for the fourth order finite difference method are missing.

Observe the following situation for the time evolution of a partial differential equation of first order in time.

$$f_t = F(f, f_x, f_{xx}, \ldots) \tag{1.9}$$

We consider a one dimensional interval $[x_{\min}, x_{\max}]$ with $N+1$ vertex-centered grid points x_i together with the function values f_i^n at these points which together represent the current time slice t^n. To calculate the function at the next slice t^{n+1} with an explicit method in time, we require the derivative(s) of the given function at the current time slice.

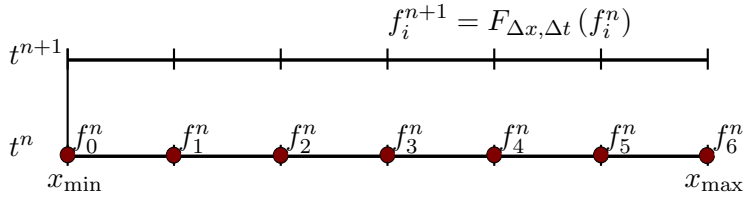


Figure 1.4: Time iteration with $N = 6$ grid cells and 7 grid points.

The problem of calculating the derivative at a grid point x_i can be solved by first calculating the Taylor polynomial $p(x)$ locally at x_i, which means that only the nearest grid points are used, and then differentiating it. So the derivative f_i' of f at x_i is given by the evaluation of the derivative of the Taylor polynomial via

$$f_i' \approx p'(x_i). \tag{1.10}$$

In this way we obtain approximations for the derivative at arbitrary points in the interval.

With this simple idea we can deduce formulas for any derivative with any order of truncation error for centered, left-sided, right-sided, centered-right-sided, centered-left-sided, etc. derivatives only depending on the choice of points near x_i and the evaluation of $p'(x)$.

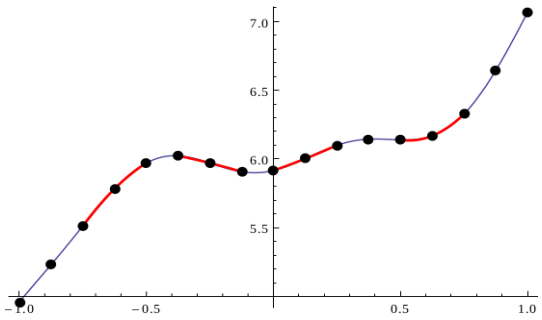


Figure 1.5: This graphic illustrates how a function is approximated locally by Taylor polynomials.

1.2.2 Second order finite difference method

To derive the formulas for the second order finite difference method we choose an expansion point $\bar{x}$ in the interval $[x_{\min}, x_{\max}]$ and denote the grid spacing by Δx. Then we form the vectors

$$\vec{x} = \begin{pmatrix} \bar{x} - \Delta x \\ \bar{x} \\ \bar{x} + \Delta x \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \qquad \vec{f} = \begin{pmatrix} f(\bar{x} - \Delta x) \\ f(\bar{x}) \\ f(\bar{x} + \Delta x) \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix} \tag{1.11}$$

which means that we choose the left and right neighbor of our expansion point together with their function values. So the interpolation problem reads for a general second order

polynomial $p_2(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2$

$$p_2(\vec{x}) = \vec{f} \qquad \text{or} \tag{1.12}$$

$$p_2(x_1) = f_1 \qquad p_2(x_2) = f_2 \qquad p_2(x_3) = f_3. \tag{1.13}$$

Solving this system gives the coefficients

$$a_0 = \frac{2f_2\Delta x^2 + (f_1 - f_3)\bar{x}\Delta x + (f_1 - 2f_2 + f_3)\bar{x}^2}{2\Delta x^2} \tag{1.14a}$$

$$a_1 = \frac{(-f_1 + f_3)\Delta x - 2(f_1 - 2f_2 + f_3)\bar{x}}{2\Delta x^2} \tag{1.14b}$$

$$a_2 = \frac{f_1 - 2f_2 + f_3}{2\Delta x^2} \tag{1.14c}$$

which leads to the following expression for the first derivative of the Taylor polynomial:

$$p_2'(x) = \frac{(-f_1 + f_3)\Delta x - 2(f_1 - 2f_2 + f_3)x_0}{2\Delta x^2} + 2\frac{f_1 - 2f_2 + f_3}{2\Delta x^2} \cdot x \tag{1.15}$$

By the insertion of $x_0 - \Delta x$, x_0 and $x_0 + \Delta x$ we get expressions for the left-sided, centered and right-sided first derivative for the second order finite difference method. The last row in the table shows the associated *stencils*, which means the geometric arrangement of grid points used in the calculation, in each of these cases:

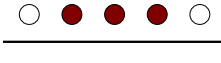
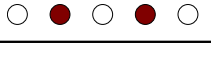
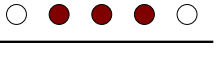
left	**centered**	**right**
$-\frac{3f_1-4f_2+f_3}{2\Delta x}$	$\frac{-f_1+f_3}{2\Delta x}$	$\frac{f_1-4f_2+3f_3}{2\Delta x}$
		
		

Table 1.1: Formulas and stencils for the first derivative with the second order finite difference method.

Next we consider the second derivative. As we loose one order of the approximating polynomial with each differentiation, we have to start with a third order polynomial to achieve second order accuracy for the second derivative. By adopting the notation of Equation 1.11 the interpolation variables read:

$$\vec{x} = \begin{pmatrix} \bar{x} - \Delta x \\ \bar{x} \\ \bar{x} + \Delta x \\ \bar{x} + 2 \cdot \Delta x \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} \qquad \vec{f} = \begin{pmatrix} f(\bar{x} - \Delta x) \\ f(\bar{x}) \\ f(\bar{x} + \Delta x) \\ f(\bar{x} + 2 \cdot \Delta x) \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix} \tag{1.16}$$

Solving $p_3(\vec{x}) = \vec{f}$ for a third order polynomial $p_3(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + a_3 \cdot x^3$ gives the

coefficients

$$a_0 = \\ = \frac{6\Delta x^3 f_2 + \Delta x^2(2f_1 + 3f_2 - 6f_3 + f_4)\bar{x} + 3\Delta x(f_1 - 2f_2 + f_3)\bar{x}^2 + (f_1 - 3f_2 + 3f_3 - f_4)\bar{x}^3}{6\Delta x^3} \tag{1.17a}$$

$$a_1 = -\frac{\Delta x^2(2f_1 + 3f_2 - 6f_3 + f_4) + 6\Delta x(f_1 - 2f_2 + f_3)\bar{x} + 3(f_1 - 3f_2 + 3f_3 - f_4)\bar{x}^2}{6\Delta x^3} \tag{1.17b}$$

$$a_2 = \frac{\Delta x(f1 - 2f_2 + f_3) + (f1 - 3f_2 + 3f_3 - f_4)\bar{x}}{2\Delta x^3} \tag{1.17c}$$

$$a_3 = \frac{-f_1 + 3f_2 - 3f_3 + f_4}{6\Delta x^3} \tag{1.17d}$$

which lead to:

$$p_3''(x) = 2\frac{\Delta x(f1 - 2f_2 + f_3) + (f1 - 3f_2 + 3f_3 - f_4)\bar{x}}{2\Delta x^3} + 2 \cdot 3\frac{-f_1 + 3f_2 - 3f_3 + f_4}{6\Delta x^3} \cdot x \tag{1.18}$$

The evaluation at the inner points $\bar{x}$ and $\bar{x} + \Delta x$ give the formula for the centered second derivative and the evaluation at the outer points the formulas for the left- and right-sided cases respectively.

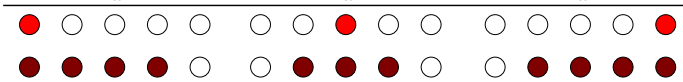
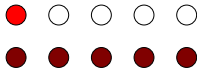
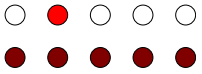
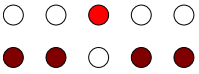
left	**centered**	**right**
$\frac{2f_1-5f_2+4f_3-f_4}{\Delta x^2}$	$\frac{f_1-2f_2+f_3}{\Delta x^2}$	$\frac{-f_1+4f_2-5f_3+2f_4}{\Delta x^2}$
		
		

Table 1.2: Formulas and stencils for the second derivative with the second order finite difference method.

1.2.3 Fourth order finite difference method

As in the case of the second order finite difference method we derive the formulas for the first derivative by solving an interpolation problem with notation analogues to Equation 1.11. Here we require a fourth order polynomial $p_4(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + a_3 \cdot x^3 + a_4 \cdot x^4$ to derive the formulas by solving $p_4(\vec{x}) = \vec{f}$.

As for the second order finite difference method, the second derivative requires an additional grid point and a polynomial order increased by one. Then the formulas are given by the following table:

left	cen.-left	centered
$\_\ \frac{25f_1-48f_2+36f_3-16f_4+3f_5}{12\Delta x}$	$\frac{-3f_1-10f_2+18f_3-6f_4+f_5}{12\Delta x}$	$\frac{f_1-8f_2+8f_4-f_5}{12\Delta x}$
		

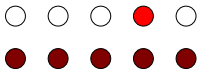
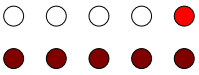
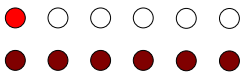
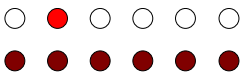
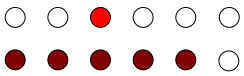
cen.-right	right
$\_\ \frac{f_1-6f_2+18f_3-10f_4-3f_5}{12\Delta x}$	$\frac{3f_1-16f_2+36f_3-48f_4+25f_5}{12\Delta x}$
	

Table 1.3: Formulas and stencils for the first derivative with the fourth order finite difference method.

left	cen.-left	centered
$\frac{45f_1-154f_2+214f_3-156f_4+61f_5-10f_6}{12\Delta x}$	$\frac{10f_1-15f_2-4f_3+14f_4-6f_5+f_6}{12\Delta x}$	$\_\ \frac{f_1-16f_2+30f_3-16f_4+f_5}{12\Delta x}$
		

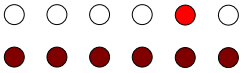
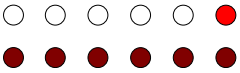
cen.-right	right
$\frac{f_1-6f_2+14f_3-4f_4-15f_5+10f_6}{12\Delta x}$	$\frac{-10f_1+61f_2-156f_3+214f_4-154f_5+45f_6}{12\Delta x}$
	

Table 1.4: Formulas and stencils for the second derivative with the fourth order finite difference method.

1.3 Spectral methods

In this section we introduce spectral methods to calculate spatial derivatives on an interval. For a detailed discussion see for example [14], where polynomial approximations are discussed in chapter two and five. This is also the main reference for the results presented in this section. In the following we state the main ideas required to use Chebyshev polynomials in numerical simulations.

1.3.1 The general Sturm-Liouville problem

We first formulate the Sturm-Liouville problem as an eigenvalue problem of the form

$$-(p \cdot u')' + q \cdot u = \lambda \cdot w \cdot u \qquad \text{on } (-1,1) \tag{1.19}$$

with suitable boundary conditions for the function u and $\lambda \in \mathbb{R}$. Here p, q and w are three given, real-valued functions satisfying:

1. p is continuously differentiable, strictly positive in $(-1,1)$ and continuous at $x = \pm 1$.
2. q is continuous, nonnegative and bounded in $(-1,1)$.
3. The *weight function* w is continuous, nonnegative and integrable over $(-1,1)$. Moreover, we assume that $\frac{1}{w}$ is integrable too.

We are interested in the expansion of functions on the interval $(-1,1)$ with respect to eigenfunctions of this problem. The reason is that one can show that these eigenfunctions form a basis for the space of functions (satisfying some special conditions) on $(-1,1)$. The aim of this subsection is to clarify this vague statement.

First of all we can distinguish between the *regular Sturm-Liouville problem* which implies that the function p is bounded from below by a positive constant and the *singular Sturm-Liouville problem* which occurs when p vanishes for at least one point on the boundary. We focus on the singular Sturm-Liouville problem because it will lead to the approximation methods used in this thesis.

We will consider only the case with $p(-1) = p(1) = 0$ and restrict our considerations in the following only to functions u in the space

$$X = \{v \in L^2_w(-1,1) \cap L^2_w(-1,1) : v' \in L^2_p(-1,1)\}. \tag{1.20}$$

Here $L^2_f(-1,1)$ denotes the Hilbert space given by the following inner product with weight function f:

$$< g, h >_f := \int_{(-1,1)} g(x) \cdot h(x) \cdot f(x)\, dx \tag{1.21}$$

So u is square integrable with respect to both the weights q and w, and u' is square integrable with respect to the weight p.

These assumptions lead to the following result stated here as a theorem:

Theorem 1.1. *With the notation and terms introduced in this section, the Sturm-Liouville problem stated in Equation 1.19 can be reformulated into a variational problem:*

$$\int_{(-1,1)} (p \cdot u' \cdot v' + q \cdot u \cdot v)\, dx = \lambda \cdot \int_{(-1,1)} u \cdot v \cdot w\, dx \quad \forall v \in X \tag{1.22}$$

Then the eigenvalues of this problem form an unbounded sequence of nonnegative real numbers $0 \leq \lambda_0 \leq \cdots \leq \lambda_k \leq \cdots$ *where each of them has finite multiplicity.*
The system of corresponding eigenfunctions ϕ_k *is orthogonal and complete in* $L^2_w(-1,1)$.
Under the sole assumption that u *be infinitely differentiable the expansion coefficients in this base decay faster than algebraically.*

Now the question arises how those eigenfunctions look like. It can be shown (and so it was done in [14, p. 279]) that the only polynomial eigenfunctions of a singular Sturm-Liouville problem are the Jacobi polynomials. More precisely they are the eigenfunctions of Equation 1.19 for

$$p(x) = (1-x)^{1+\alpha} \cdot (1+x)^{1+\beta} \tag{1.23a}$$

$$q(x) = 0 \tag{1.23b}$$

$$w(x) = (1-x)^{\alpha} \cdot (1+x)^{\beta} \tag{1.23c}$$

where α, $\beta > -1$ with corresponding eigenvalues for degree k:

$$\lambda_k = k(k+\alpha+\beta+1) \tag{1.24}$$

The Jacobi polynomials take with the normalization $P_k^{(\alpha,\beta)}(1) = \binom{k+\alpha}{k}$ the form

$$P_k^{(\alpha,\beta)}(x) = \frac{1}{2^k} \sum_{l=0}^{k} \binom{k+\alpha}{l} \binom{k+\beta}{k-l} (x-1)^l (x+1)^{k-l}. \tag{1.25}$$

Jacobi polynomials for which $\alpha = \beta$ are called *ultraspherical polynomials* and are denoted by $P_k^{(\alpha)}(x)$. For special choices for α we arrive at the well known polynomial families stated in the following table:

Name	$\boldsymbol{\alpha}$	**Expression**
Legendre polynomials	0	$L_k(x) = P_k^{(0)}(x)$
Chebyshev polynomials (1st kind)	$-\frac{1}{2}$	$T_k(x) = P_k^{\left(-\frac{1}{2}\right)}(x) \diagup P_k^{\left(-\frac{1}{2}\right)}(1)$
Gegenbauer polynomials	$\nu - \frac{1}{2}$	$C_k^{\nu}(x) = \frac{\Gamma(\nu+\frac{1}{2})\Gamma(2\nu+k)}{\Gamma(\nu+k+\frac{1}{2})\Gamma(2\nu)} P_k^{\left(\nu-\frac{1}{2}\right)}(x)$

Table 1.5: Definitions of various ultraspherical polynomial families.

1.3.2 General properties of orthogonal systems of polynomials

Next we discuss properties and techniques used to expand functions in terms of orthogonal systems of polynomials as for example the ones introduced in Table 1.5.

Let $\mathbb{P}_N$ denote the space of all polynomials of order $\leq N$. Furthermore let $\{p_k\}_{k\in\mathbb{N}_0}$ denote a system of algebraic polynomials with orders k satisfying

$$< p_k, p_m >_w = \int_{(-1,1)} p_k(x) \cdot p_m(x) \cdot w(x)\, dx = 0 \qquad \text{for } m \neq k. \tag{1.26}$$

Here the inner product is defined as in Equation 1.21. Hence $\{p_k\}_{k\in\mathbb{N}_0}$ is an orthogonal

system over the interval $(-1,1)$. We already mentioned that the systems in Table 1.5 are complete in $L^2_w(-1,1)$.

Now we consider the expansion of a function $u \in L^2_w(-1,1)$ in terms of the system $\{p_k\}_{k\in\mathbb{N}_0}$ which reads:

$$u = \sum_{k=0}^{\infty} \hat{u}_k \cdot p_k \tag{1.27}$$

The expansion coefficients $\hat{u}_k$ are given by the orthogonal projection related to the inner product of $L^2_w(-1,1)$ via

$$\hat{u}_k = \frac{1}{||p_k||^2_w} \int_{(-1,1)} u(x) \cdot p_k(x) \cdot w(x)\, dx \tag{1.28}$$

where $||.||^2_w$ denotes the norm associated to the inner product and the necessary normalization factors are incorporated in the definition of $\hat{u}_k$. We call the set of all $\hat{u}_k$ obtained by Equation 1.28 the *polynomial transform of u*.

By choosing an integer $N > 0$ we may truncate the polynomial series and get the polynomial

$$P_N u = \sum_{k=0}^{N} \hat{u}_k \cdot p_k. \tag{1.29}$$

This is nothing but the projection of u to the space $\mathbb{P}_N$ via the inner product of Equation 1.21. So

$$< P_N u, v >_w =< u, v >_w \qquad \forall v \in \mathbb{P}_N. \tag{1.30}$$

By the completeness of $\{p_k\}_{k\in\mathbb{N}_0}$ it follows that for all $u \in L^2_w(-1,1)$ we get

$$\lim_{N\to\infty} ||u - P_N u||^2_w = 0. \tag{1.31}$$

1.3.3 Chebyshev polynomials

In this thesis we will focus on Chebyshev polynomials (or more precisely Chebyshev polynomials of the first kind) because they have certain properties which turn out to be advantageous during the simulation process of our numerical problems later on.

First we discuss their representation. As mentioned above we get them via a singular Sturm-Liouville problem by choosing in Equation 1.19 $p(x) = (1-x^2)^{\frac{1}{2}}$, $q(x) = 0$ and $w(x) = (1-x^2)^{-\frac{1}{2}}$ which leads with the eigenvalues $\lambda_k = k^2$ to:

$$\left(\sqrt{1-x^2} \cdot T_k'(x)\right)' + \frac{k^2}{\sqrt{1-x^2}} \cdot T_k(x) = 0 \qquad k \in \mathbb{N}_0 \tag{1.32}$$

In Table 1.5 we had a trivially normalized (i.e. $T_k(1) = 1$) representation of T_k in terms of

Jacobi polynomials. There exists a much simpler expression in trigonometric terms given by

$$T_k(x) = \cos(k\theta) \qquad \text{with} \quad \theta = \arccos(x). \tag{1.33}$$

We can check that this representation satisfies Equation 1.32 on $(-1, 1)$:

$$T_k'(x) = -k\sin(k\theta) \cdot \frac{-1}{\sqrt{1-x^2}} \tag{1.34}$$

which leads to

$$(k\sin(k\theta))' + \frac{k^2}{\sqrt{1-x^2}} \cdot T_k(x) = 0 \tag{1.35}$$

and

$$-\frac{k^2}{\sqrt{1-x^2}}\cos(k\theta) + \frac{k^2}{\sqrt{1-x^2}} \cdot T_k(x) = 0. \tag{1.36}$$

The trigonometric representation satisfies $T_k(1) = 1$ too as $\arccos(1) = 0$. This is not enough to show the equivalence of the two representations, because Equation 1.32 is not Lipschitz at $x = 1$. We will not go into detail here but refer to the large amount of available literature on this subject.

Some properties of the Chebyshev polynomials are immediately clear from the trigonometric representation:

$$|T_k(x)| \leq 1 \tag{1.37a}$$

$$T_k(\pm 1) = (\pm 1)^k \tag{1.37b}$$

where the latter one follows from $\arccos(1) = 0$ and $\arccos(-1) = \pi$.

By Equation 1.27 the expansion of a function $u \in L_w^2(-1, 1)$ in terms of Chebyshev polynomials is given by

$$u = \sum_{k=0}^{\infty} \hat{u}_k \cdot T_k \qquad \text{with} \quad \hat{u}_k = \frac{1}{||T_k||_w^2} \int_{(-1,1)} u(x) \cdot T_k(x) \cdot w(x)\, dx. \tag{1.38}$$

Equivalently we can define the function $\bar{u}(\theta) = u(\cos(\theta))$, allowing us to use the expansion

$$\bar{u}(\theta) = \sum_{k=0}^{\infty} \hat{u}_k \cdot \cos(k\theta) \tag{1.39}$$

which shows that the Chebyshev series for u corresponds to a cosine series for $\bar{u}$. This is one of the main advantages of the Chebyshev polynomials. With this correspondence a number of techniques known for Fourier series can be applied to Chebyshev polynomials. In particular, the *Fast Fourier transformation FFT* becomes applicable which allows a faster change between spatial and spectral representations of a function.

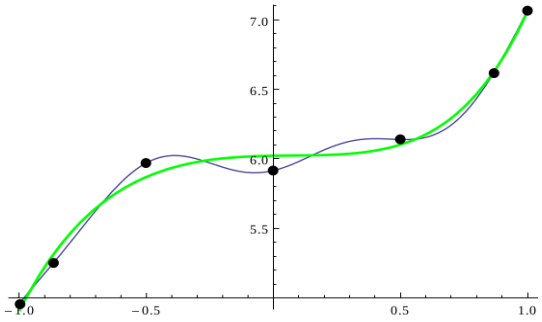


Figure 1.6: This graphic illustrates how a function is approximated globally by a Chebyshev series.

1.3.4 Interpolating versus non-interpolating methods

Before we start a more detailed discussion of the realization of spectral approximation and differentiation methods for Chebyshev polynomials, we take a short break and clarify some terminology following [7, p. 12].

We saw in Equation 1.28 that for the calculation of the spectral coefficients $\hat{u}_k$ we need to evaluate an integral. The way how this problem is solved makes the difference between *interpolating or pseudospectral* and *non-interpolating or integration methods*.

In the pseudospectral case we choose grid points in the interval where we want to expand the function u. Then we require that the truncated series $P_N u$ agrees with the function u on the previously chosen grid points. This gives a system of equations which allow the calculation of the spectral coefficients if the number of grid points corresponds to the number of coefficients to be calculated. These special grid points are called *collocation* or *interpolation* points.

In the other case of non-interpolating methods we do not require a special choice of grid points. Instead the function u is multiplied with a given basis function as in Equation 1.28 and an arbitrary integration algorithm may be chosen to calculate the spectral coefficients.

It should be mentioned that although the integration process may be arbitrary in the non-interpolating case, there exist natural choices for each basis set by taking into account a fast computability and an as small as possible error in the approximation. So there exist situations were both cases lead to equivalent results.

1.3.5 Collocation with Chebyshev polynomials

One of the well known problems regarding the approximation of functions by polynomials is the so called Runge phenomenon discussed in [7, p. 83] which was already introduced in [26]. This phenomenon describes that polynomial approximations on an equidistant grid may lead to bigger errors when increasing the order of approximation. The reason for that are oscillations in the approximation which occur at the boundary of the interval and soon exceed

the maximum values of the function itself. The general idea is to abandon equidistant grid spacing and use a distribution of grid points which increases their density near the boundary.

A list of typical collocation points for the Chebyshev polynomials is given in [14, p. 85]. Table 1.6 contains illustrations of the collocation points for $N = 6$.

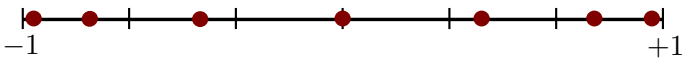
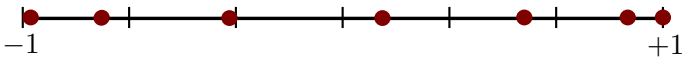
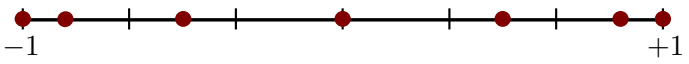
Name	**Formula**
Chebyshev Gauss	$x_i = \cos\left(\frac{(2i+1)\pi}{2N+2}\right) \quad i = 0, \ldots, N$ 
Chebyshev Gauss-Radau	$x_i = \cos\left(\frac{2\pi i}{2N+1}\right) \quad i = 0, \ldots, N$ 
Chebyshev Gauss-Lobatto	$x_i = \cos\left(\frac{\pi i}{N}\right) \quad i = 0, \ldots, N$ 

Table 1.6: Typical collocation points for Chebyshev polynomials.

The most important difference between the three choices is the position of the outermost points. While Chebyshev Gauss and Chebyshev Gauss-Radau points use none or only one boundary point of the interval, the Chebyshev Gauss-Lobatto points use both. Which collocation choice fits best depends on practical purposes, e.g. the implementation of the boundary conditions.

We choose the Chebyshev Gauss-Lobatto collocation points for our further considerations and numerical simulations. The reason is the availability of collocation points at the interval boundaries for easier boundary condition formulations. By inserting their definition into Equation 1.33, we observe that these collocation points correspond to the extremal points of the Chebyshev polynomials.

We note that Chebyshev Gauss-Radau points might be advantageous when one tries to avoid singularities at the origin in spherical symmetric problems with purely pseudospectral methods.

1.3.6 Transformation between function values and spectral coefficients

Equation 1.27 shows how the spectral coefficients of Chebyshev (or arbitrary) polynomials lead to appropriate function values on the interval $[-1, 1]$. If we restrict our considerations to the collocation points x_{col} in this interval, the summation becomes a linear transformation $\mathcal{T}_N$ from the $\mathbb{R}^{N+1}$ of coefficients for the first $N + 1$ Chebyshev polynomials C_{Cheb} to the

$\mathbb{R}^{N+1}$ of function values at the collocation points $\mathrm{F_{Col}}$:

$$\mathcal{T}_N : \mathrm{C_{Cheb}} \to \mathrm{F_{Col}} \qquad \mathcal{T}_N(\hat{u}) = \mathcal{T}_N \cdot \hat{u} = \sum_{k=0}^{N} \hat{u}_k \cdot T_k \mid_{x_{\mathrm{col}}} \tag{1.40}$$

So the matrix $\mathcal{T}_N$ (there is no need to distinguish between the mapping and the matrix in this context) consists of the Chebyshev polynomials evaluated at the collocation points where the column index corresponds to the polynomial order and the row index to the collocation point. This gives in the trigonometric version the following expression for the matrix coefficients:

$$(\mathcal{T}_N)_{ik} = \cos\left(\frac{\pi i k}{N}\right) \tag{1.41}$$

To get the inverse transformation one could simply invert the matrix $\mathcal{T}_N$ via a built-in function of the simulation tool or, if one is concerned about possible numerical errors arising through this process for larger matrices, calculate the inverse by applying the following formula

$$(\mathcal{T}_N^{-1})_{ki} = \frac{2}{N \bar{c}_i \bar{c}_k} \cos\left(\frac{\pi i k}{N}\right) \tag{1.42}$$

where $\bar{c}_0 = 2$, $\bar{c}_N = 2$ and the others are equal to 1.

This illustrates the accumulated advantages of Chebyshev polynomials, Chebyshev Gauss-Lobatto collocation and the trigonometric representation. All expressions take a very simple form compared to the general case of Jacobi polynomials in Equation 1.25 and are the first choice in case there are no significant reasons for alternatives. Moreover both transformations may be evaluated by the Fast Fourier Transformation.

1.3.7 Differentiation in spectral representation

From the representation of a function u as a sum of Chebyshev polynomials given by Equation 1.38 it follows that the derivative of this function reads

$$u' = \sum_{k=0}^{\infty} \hat{u}_k \cdot T_k' \tag{1.43}$$

where T_k' is given by Equation 1.34.

Following [5, p. 220] we can expand the derivative T_k' again in the base of Chebyshev polynomials via Equation 1.38:

$$u' = \sum_{k=0}^{\infty} \hat{u}_k \cdot \sum_{l=0}^{\infty} D_{lk} \cdot T_l \tag{1.44}$$

The coefficients D_{lk} are again given by the integral in Equation 1.38:

$$D_{lk} = \frac{1}{||T_l||_w^2} \int_{(-1,1)} T_k(x) \cdot T_l(x) \cdot w(x)\, dx \tag{1.45}$$

It turns out that these coefficients have integer values which can be seen by inserting the trigonometric expressions for small values or using trigonometric relations (or a computer algebra system) to calculate the coefficients for the ranges required for the simulations.

$$D^{(1)} = \begin{pmatrix} 0 & 1 & 0 & 3 & 0 & 5 & \cdots \\ 0 & 0 & 4 & 0 & 8 & 0 & \cdots \\ 0 & 0 & 0 & 6 & 0 & 10 & \cdots \\ 0 & 0 & 0 & 0 & 8 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & 10 & \cdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \tag{1.46}$$

Even if we use trigonometric expressions for calculation, we are still using polynomials. So its clear that the lower left part of the coefficient matrix remains zero, because we do not need higher order polynomials to express derivatives of lower order polynomials. This leads to the following representation for the derivative of a function:

$$u' = \sum_{k=0}^{\infty} \sum_{l=0}^{\infty} \hat{u}_k \cdot D_{lk} \cdot T_l = \sum_{l=0}^{\infty} \hat{u}'_l \cdot T_l \quad \text{with } \hat{u}'_l = \sum_{k=0}^{\infty} \hat{u}_k \cdot D_{lk} \tag{1.47}$$

We call $D^{(1)}$ our *spectral derivative matrix*. Because we work with a finite expansion, we denote the spectral derivative matrix with $N+1$ rows and columns by $D_N^{(1)}$.

The matrix for the second derivative is given by the product of two matrices for the first derivative. The analog is true for derivatives of higher order. All the coefficients stay integers.

1.3.8 Differentiation in space representation

In order to get a method for the differentiation of a function in space representation, we only have to combine the previously introduced methods.

$$\mathcal{T}_N \circ D_N^{(1)} \circ (\mathcal{T}_N^{-1}) : \mathrm{F_{Col}} \to \mathrm{C_{Cheb}} \to \mathrm{C_{Cheb}} \to \mathrm{F_{Col}} \tag{1.48}$$

Analogously, higher order differentiation matrices for functions in space representation are created by additional applications of the first derivative matrix between the transformations. This does not lead to additional errors compared to a direct calculation of the higher spectral derivative matrices because all entries are integers as mentioned above.

To save resources and computation time, the whole calculation is done only once in the

initialization process. The resulting matrix is saved and used during the simulation process.

1.4 Mesh refinement and the hybrid grid approach

In general, numerical simulations require a broad range of scales to be resolved. Therefore, only using one grid to discretize the physical domain may not be an efficient strategy.

In the case of an isolated, asymptotically flat system we can identify two regions with special character. The first one lies near the black hole horizon(s) and is characterized by large curvature. The second one is the wave zone, where the spacetime is almost flat. For binary black hole systems gravitational waves are extracted at a distance on the order of hundred times the system mass from the source.

In [6] the authors introduced the so called *adaptive mesh refinement method* for hyperbolic partial differential equations and finite difference grids, where the resolution in specific regions is set based on the estimated truncation error. For applications of related *fixed mesh refinement* methods in numerical relativity see [24, 13, 2, 11, 12]. We will not discuss general methods and properties used in this context. We will rather focus on one dimensional domains and give some motivation for a *hybrid grid approach* that combines finite difference and spectral methods as defined below.

1.4.1 Motivation

The decomposition of the physical domain into subdomains depends on what we expect the solution to look like there. For example, there might be regions where the solution of a differential equation stays to some extent constant whereas in other regions large gradients occur. In such a situation it would be inconvenient to discretize the later region with an grid as coarse as the first one. On the other hand, an approximation via a very fine grid for a constant function leads to a waste of resources and computation time. This would be a typical application for a mesh refinement strategy mentioned above.

Another important consideration is the implementation of boundary conditions which depends on the type of grid used. Imposing Dirichlet boundary conditions is very simple to handle with a vertex-centered grid whereas, in case of a radial coordinate, the right hand side of some solution components at the center might diverge due to the coordinate singularity inherent in spherical coordinates, what makes a cell-centered grid a better choice.

1.4.2 The hybrid grid approach

Now we turn to the discussion of what we call the *hybrid grid approach*. We decompose the physical domain (in our case an interval) into patches with not only different resolutions but different approximation methods. We can combine (any number of) patches where we use either the finite difference method on uniform (vertex-centered or cell-centered) grids, or

the Chebyshev pseudospectral method on collocation grids. The aim of this approach is to combine the advantages of both methods.

One of our aims will be to apply the hybrid grid to the GBSSN system in spherical symmetry with wormhole puncture initial data (see Section 6.1), as it was done in [8] for the finite difference method only. The simulations discussed there did not use filtering techniques and where therefore limited to run-times of some hundred mass-time intervals.

In [16] purely pseudospectral methods were used and a region inside the horizon was excised. Although one does not have to take care of singularities near the origin, the *excision methods* lead to additional problems if one considers moving punctures, e.g. in case of binary black hole simulations the horizons have to be tracked throughout the simulation. Nevertheless this approach leads to run-times of approximately $t = 10000M$ and more.

Another approach introduced in [10] uses the so called technique of *turduckening*, i.e. some smooth, unphysical but numerically well behaving data is inserted into the region of the black hole interior to avoid the singular behavior.

The aim of the hybrid grid approach is to use the full wormhole puncture initial data (without excision or turduckening) and assess possible improvements gained by the introduction of a pseudospectral grid farther away from the black hole horizon.

The main issue for the realization of an accurate and stable hybrid grid method is the *interface* between the finite difference (FD) region, where we use local approximations of the evaluated functions, and the pseudospectral (PS) one, where we have a global expansion in a polynomial basis over the whole patch. Our aim will be to formulate these interface conditions with the help of characteristic eigenfields of the various partial differential equations under consideration.

Throughout this thesis we will consider one-dimensional or spherical symmetric problems only. Hence the already mentioned grid patches will actually be one-dimensional grid segments. Nevertheless the hybrid grid method does not depend on this restriction. Also in more than one dimension the key ideas should be applicable.

1.5 Method of Lines (MOL)

The simulation of partial differential equations (PDEs) requires a discretization of all variables. The PDEs in this thesis are hyperbolic systems depending on the time variable t and on one space variable x (Cartesian coordinate) or r (radial coordinate). In principle, it would be possible to use the same methods for the discretization in space and time as multidimensional methods can handle for example functions depending on two or three space dimensions. It turns out to be convenient for practical and intuitive reasons to treat space and time coordinates differently.

In the *method of lines (MOL)* (see [5, p. 208]), we first discretize the space coordinates only. There is no reason to distinguish between finite difference, pseudospectral or the hybrid grid

methods at this stage. The system of partial differential equations for the function $u(t, x)$ then becomes a set of ordinary differential equations for the functions $u_j(t, x_j)$, i.e. an ordinary differential equation for each grid point respectively.

The next step is the integration of all these ordinary differential equations. Here lies the big advantage of MOL: We are totally free to choose an appropriate algorithm for the time integration independent of the spatial discretization. We do not have to care if we used finite difference or pseudospectral methods, and therefore this is perfectly applicable to the hybrid approach.

In our later simulations we use the fourth-order Runge-Kutta (RK4) method described in [14, p. 524]. This is a single-step, multistage time discretization method. This means that the next time step depends only on the previous one and through the calculation of this step the right hand side of the ordinary differential equations is applied multiple times to increase accuracy. So this ODE integration method is perfectly well suited for the situation of initial data on a single time slice.

For an ordinary differential equation

$$\frac{\partial f}{\partial t} = \mathrm{rhs}(f, t) \tag{1.49}$$

the algorithm reads:

$$k_1 = \mathrm{rhs}\left(f_i, t^n\right) \tag{1.50a}$$
$$k_2 = \mathrm{rhs}\left(f_i + 0.5 \cdot \mathrm{dt} \cdot k_1, t^n + 0.5 \cdot \mathrm{dt}\right) \tag{1.50b}$$
$$k_3 = \mathrm{rhs}\left(f_i + 0.5 \cdot \mathrm{dt} \cdot k_2, t^n + 0.5 \cdot \mathrm{dt}\right) \tag{1.50c}$$
$$k_4 = \mathrm{rhs}\left(f_i + \mathrm{dt} \cdot k_3, t^n + \mathrm{dt}\right) \tag{1.50d}$$
$$f_{i+1} = f_i + \frac{1}{6} \cdot \mathrm{dt} \cdot (k_1 + 2k_2 + 2k_3 + k_4) \tag{1.50e}$$

1.6 Numerical convergence

This thesis is not the place for a detailed treatment of numerical convergence. Nevertheless we will introduce the major terms used in the following. For more details see for example [7, p. 25ff.] and [19, p. 381, p. 465ff.] which also act as references for this section.

1.6.1 Preliminary definitions

In this section we will consider partial differential equations of the form:

$$\partial_t u = P\left(x, t, \partial_x\right) \cdot u + F, \qquad 0 \leq x \leq l, \quad t \geq t_0 \tag{1.51a}$$
$$u(x, t_0) = f(x) \tag{1.51b}$$

Here $u = \left(u^{(1)}, \ldots, u^{(m)}\right)^T$ is a vector function with m components and

$$P\left(x, t, \partial_x\right) = \sum_{\nu=0}^{p} A_\nu(x,t) \frac{\partial^\nu}{\partial x^\nu} \tag{1.52}$$

is a differential operator of order p with smooth matrix coefficients. At $x = 0$ and $x = l$ we give boundary conditions

$$L_0(t, \partial_x)u(0,t) = g_0 \qquad L_1(t, \partial_x)u(l,t) = g_1. \tag{1.53}$$

Here L_0 and L_1 are differential operators of order r. In most applications $r \leq p - 1$.

Definition 1.2. Consider the above partial differential equation with $F = g_0 = g1 = 0$. We call the problem *well posed* if, for every smooth f that vanishes in a neighborhood of $x = 0$ and $x = l$, it has a unique smooth solution that satisfies the estimate

$$||u(.,t)|| \leq K \cdot e^{\alpha(t-t_0)} ||u(.,t_0)||, \tag{1.54}$$

where K and α do not depend on f and t_0.

After the introduction of a grid, grid functions and a discrete norm $||\,.\,||_h$, the continuous problem can be approximated by

$$\frac{dv_j}{dt} = Q(x_j, t, D)v_j + F_j, \qquad j = 1, 2, \ldots, N-1, \quad t \geq t_0 \tag{1.55a}$$

$$v_j(t_0) = f_j, \tag{1.55b}$$

$$L_0(t, D)v_0(t) = g_0(t), \qquad t \geq t_0, \tag{1.55c}$$

$$L_N(t, D)v_N(t) = g_N(t), \qquad t \geq t_0. \tag{1.55d}$$

Here D is an arbitrary difference operator in the x direction. For convenience, we have used the same notation for the grid functions F_j, f_j and g_0 as used for the corresponding functions in the continuous problem, even though they may be different in the grid points. We assume that the grid and the boundary conditions are such that v is uniquely determined.

Definition 1.3. We consider now the approximating system with $F = g_0 = g_N = 0$. The approximation is called *stable* if, for all $h \leq h_0$, there are constants K and α such that, for all t_0 and all $v(t_0)$,

$$||v(t)||_h \leq K \cdot e^{\alpha(t-t_0)} ||v(t_0)||_h. \tag{1.56}$$

The constants K and α are, in general, different from the corresponding ones for the continuous problem. The assumption of a unique solution implies that f_j be finite for every j and every fixed h.

Definition 1.4. The approximation is strongly stable if it is stable and if the estimate

$$||v(t)||_h \leq K(t,t_0)\left(||v(t_0)||_h^2 + \max_{t_0\leq\tau\leq t}||F(\tau)||_h^2 + \max_{t_0\leq\tau\leq t}\left(|g_0(\tau)|^2 + |g_N(\tau)|^2\right)\right) \tag{1.57}$$

holds. Here $K(t,t_0)$ is a bounded function in any finite time interval and does not depend on the data.

1.6.2 Orders of convergence

Throughout our numerical experiments we will require a method to check the implementation of various differentiation methods. There we will measure the errors arising and compare the rate of their decrease to the theoretical optimum of the method. We will talk about convergence rates of the methods. To clarify what is meant with order of convergence, we state the following definitions:

Definition 1.5. Let $a = \sum_{n=0}^{\infty} a_n$ denote a series. The *algebraic index of convergence* k is the largest number for which

$$\lim_{n\to\infty} |a_n| \cdot n^k < \infty, \quad n >> 1 \tag{1.58}$$

where the a_n are the coefficients of the series.

Alternatively: If the coefficients of a series are a_n and if

$$a_n \sim \mathrm{O}\left(\frac{1}{n^k}\right), \quad n >> 1 \tag{1.59}$$

then k is the algebraic index of convergence.

Definition 1.6. If the algebraic index of convergence is unbounded, which means that the coefficients a_n decrease faster than n^{-k} for any finite power of k, then the series is said to have the property of *infinite order*, *exponential* or *spectral convergence*.

Alternatively: If

$$a_n \sim \mathrm{O}\left(\exp(-q \cdot n^r)\right), \quad n >> 1 \tag{1.60}$$

with q a constant for some $r > 0$, then the series has *infinite order* or *exponential convergence*.

Example 1.7. Before we start further discussions, we remember the previously mentioned Chebyshev series. This is an infinite series of polynomials. By expanding a given function f in terms of Chebyshev polynomials T_k, we get a list of expansion coefficients c_n.

$$f(x) = \sum_{n=0}^{\infty} c_n \cdot T_n(x) \tag{1.61}$$

Let now $a = f(x_0)$ denote the function value of f at a given point x_0. Then we get a series

for the calculation of this function value via

$$a = f(x_0) = \sum_{n=0}^{\infty} c_n \cdot T_n(x_0) = \lim_{N\to\infty} \sum_{n=0}^{N} c_n \cdot T_n(x_0) = \lim_{N\to\infty} \sum_{n=0}^{N} a_n \tag{1.62}$$

where $a_n = c_n \cdot T_n(x_0)$. So we found an example of a series as discussed above by considering the pointwise approximation of a function value.

It should be emphasized that those definitions are made asymptotically for large n. The application to small n may be misleading.

Now we turn to a description better suited for our needs.

Definition 1.8. Let $u(x,t)$ be a smooth solution of the partial differential equation. The approximation is *accurate of order p*, if the restriction to the grid satisfies the perturbed system

$$\frac{du_j}{dt} = Q(x_j,t,D)u_j + F_j + h^p\tilde{F}_j, \qquad j = 1,2,\ldots,N-1, \tag{1.63a}$$

$$u_j(t_0) = f_j + h^p\tilde{f}_j, \tag{1.63b}$$

$$L_0(t,D)u_0(t) = g_0(t) + h^p\tilde{g}_0(t), \tag{1.63c}$$

$$L_N(t,D)u_N(t) = g_N(t) + h^p\tilde{g}_N(t), \tag{1.63d}$$

where $\tilde{F}_j$, $\tilde{f}_j$, $\tilde{g}_0$, $\tilde{g}_N$, $\frac{d\tilde{g}_0}{dt}$ and $\frac{d\tilde{g}_N}{dt}$ are bounded independent of h. If $p \geq 1$, then the approximation is called *consistent*.

We observe that convergence of the solutions of consistent approximations follows immediately from strong stability.

Theorem 1.9. *If the approximation is strongly stable and accurate of order p, then*

$$||u(.\,,t) - v(t)||_h \leq \text{const} \cdot h^p \tag{1.64}$$

for smooth solutions $u(x,t)$ on any finite time interval $(0,T)$.

A similar theorem for approximations that are stable but not strongly stable would require a more complicated discussion.

The importance for practical purposes of the terms introduced in this section becomes clear by considering the following theorem for linear partial differential equations.

Theorem 1.10. *(Lax-Richtmyer equivalence theorem) If the finite difference approximation to a linear partial differential equation is stable and consistent, then we obtain convergence even if the underlying continuous problem only has a generalized solution. If the approximation is convergent, then it is stable.*

We observe, that in this case convergence is equivalent to consistency and stability. Note that in most cases it is much easier to test for consistency and stability than for convergence directly.

Also note, that each computer has only the ability to compute floating point operations with finite precession. By the *machine sized values* or *machine epsilon* we mean an upper bound for the relative errors arising due to the limitation of saving the result of a floating point operation within a storage location of finite size. This implies that after reaching this physical limit, a further evaluation of convergence is not possible anymore.

1.6.3 Convergence factor

Another problem which arises through the discussion of convergence is how one should determine the error of the approximation. Of course, if an analytic solution to the problem is known, one can compare the numerical results to it. But this can only aid as test cases for the evaluation of the approximation method. The interesting problems in numerical simulations are the ones where no analytic solution is known. We require a method which at least shows that something is wrong in the simulation code, even if it is for itself not enough to prove that the numerical solution converges to the real one.

The idea is to compare the numerical results of three simulation runs with different approximation orders to an unknown analytic solution. Because the expected convergence orders of the methods are known, the analytic solution can be eliminated from the system of equations and we get *convergence factors* which relate the differences between the simulation runs. If the numerical results do not show the expected factors the reason might likely be an error in the implementation.

We consider the following system where f_{low}, f_{med}, f_{high} describe the approximations to the analytic solution f_{anl} and the factors k_{ml} and k_{hm} are determined by the method:

$$(f_{low} - f_{anl}) = k_{ml} \cdot (f_{med} - f_{anl}) \tag{1.65a}$$

$$(f_{med} - f_{anl}) = k_{hm} \cdot (f_{high} - f_{anl}) \tag{1.65b}$$

After calculating an explicit expression for f_{anl}

$$f_{anl} = \frac{f_{med} \cdot k_{ml} - f_{low}}{k_{ml} - 1} \tag{1.66}$$

we can insert it into the second equation to arrive at

$$f_{med} + f_{low} \cdot (k_{hm} - 1) + f_{high} \cdot (k_{ml} - 1) \cdot k_{hm} - f_{med} \cdot k_{ml} \cdot k_{hm} = 0 \tag{1.67}$$

respectively:

$$(f_{med} - f_{low}) - (f_{med} - f_{low}) \cdot k_{hm} - (f_{high} - f_{med}) \cdot k_{hm} + (f_{high} - f_{med}) \cdot k_{ml} \cdot k_{hm} = 0 \tag{1.68}$$

This leads to

$$(f_{med} - f_{low}) = C_f \cdot (f_{high} - f_{med}) \quad \text{with } C_f = \frac{(k_{ml} - 1) \cdot k_{hm}}{k_{hm} - 1} \tag{1.69}$$

where C_f is called the *convergence factor*. The following table gives the convergence factors for different approximation methods. The variables n_l, n_m and n_h denote the number of points on a finite difference grid or the order of the approximating polynomials respectively.

Approximation method	$\mathbf{k_{ml}}$	$\mathbf{k_{hm}}$	$\mathbf{C_f}$
Second order finite difference method	$\left(\frac{n_m}{n_l}\right)^2$	$\left(\frac{n_h}{n_m}\right)^2$	$\frac{({n_m}^2 - {n_l}^2)\cdot {n_h}^2}{({n_h}^2 - {n_m}^2)\cdot {n_l}^2}$
Fourth order finite difference method	$\left(\frac{n_m}{n_l}\right)^4$	$\left(\frac{n_h}{n_m}\right)^4$	$\frac{({n_m}^4 - {n_l}^4)\cdot {n_h}^4}{({n_h}^4 - {n_m}^4)\cdot {n_l}^4}$
Chebyshev pseudospectral method	$\exp\left(\frac{n_m}{n_l}\right)$	$\exp\left(\frac{n_h}{n_m}\right)$	$\frac{\exp\left(\frac{n_m}{n_l}+\frac{n_h}{n_m}\right)-\exp\left(\frac{n_h}{n_m}\right)}{\exp\left(\frac{n_h}{n_m}\right)-1}$

Table 1.7: Convergence factors for different approximation methods.

1.7 Filtering

Filtering in connection to numerical simulations means: dealing with usually higher frequency disturbances which cause numerical errors. Normally this happens due to strong non-linearities in the equations or imperfect boundary conditions. The literature about this topic is tremendous and we describe in the following only the main ideas and techniques used in our simulations. We refer to [1, p. 343 ff.] and [14, p. 56 ff.] for further details.

1.7.1 Artificial dissipation in the FD grid

While working with non-linear systems of equations it turns out that many methods like finite difference become slightly unstable because of both, coefficients that depend on the dynamical variables and lower order terms. These instabilities lead to high frequency oscillations.

In [29, p. 363] partial differential equations with various orders of spatial derivatives are discussed. The term *dissipation of solutions of PDEs* is defined as the behavior that Fourier modes do not grow in time and at least one mode decays, i.e. high frequency oscillations are not amplified. Additionally one defines *dispersion of solutions of PDEs* as when Fourier modes of different wave lengths propagate at different speeds.

It is observed that PDEs containing only even ordered spatial derivatives will behave dissipative, whereas PDEs containing only odd ordered spatial derivatives will behave dispersive if the order is greater than one.

The idea is to add dissipative terms to the finite difference operators that act as "low pass filters" in the sense that they preferably damp modes with wavelength similar to the grid

spacing. Such high frequency modes are already unresolved and are therefore severely affected by truncation error. Furthermore, they are also frequently the source of the instabilities. In many applications we are better off dissipating them away as otherwise they might become unstable and ruin the simulation. Adding such dissipative terms to a numerical scheme goes under the name of *artificial dissipation*. The standard way of doing this is known as *Kreiss-Oliger dissipation*.

We assume a finite difference scheme like the one given by the formula

$$u_m^{n+1} = u_m^n + \Delta t \cdot S\left(u_m^n\right) \tag{1.70}$$

where $S\left(u^n\right)$ denotes some spatial finite difference operator. We add an additional term to the right hand side.

$$u_m^{n+1} = u_m^n + \Delta t \cdot S\left(u_m^n\right) - \epsilon \cdot \frac{\Delta t}{\Delta x}(-1)^N \Delta_x^{2N}\left(u_m^n\right) \tag{1.71}$$

Here $\epsilon > 0$, $N \geq 1$ an integer and $\Delta_x^{2N}\left(u_m^n\right)$ denotes the $2N$ centered finite difference operator (without division by the appropriate power of the grid spacing). This expression can be recast into the following form:

$$\frac{u_m^{n+1} - u_m^n}{\Delta t} = S\left(u_m^n\right) - \epsilon \cdot \frac{1}{\Delta x}(-1)^N \Delta_x^{2N}\left(u_m^n\right) \tag{1.72}$$

Assuming small Δt and Δx this expression can be replaced by

$$\partial_t u = S\left(u\right) - \epsilon \cdot \Delta x^{2N-1}(-1)^N \partial_x^{2N} u \tag{1.73}$$

where for the limit of Δx going to zero we obtain the original differential equation.

The number $2N$ defines the order of the derivative which is added. It has to be at least two orders higher than the highest derivative occuring on the right hand side $S\left(u\right)$. Typically, for a second order system one chooses a fourth derivative.

How small the factor ϵ has to be chosen is a matter of trial and error, although an approximation where one can start from can be determined by methods suggested in [1, p. 344]. Terms for the boundary of the interval may be added or just set to zero.

We will use artificial dissipation for the long time simulations only. In these cases we add a fourth order derivative of the evolved variables multiplied by a small factor to our second order system.

1.7.2 Exponential filter in the PS grid

As discussed in Section 1.3 the pseudospectral method relies on the expansion of the solution in terms of polynomials. Due to limited precision the series is truncated at some order. In

Equation 1.29 this was denoted by

$$P_N u = \sum_{k=0}^{N} \hat{u}_k \cdot p_k. \tag{1.74}$$

Following [14, p. 56] this truncated series exhibits some oscillations about the exact solution. This behavior is called *Gibbs Phenomenon* and leads to instabilities. The process of damping such high frequency oscillations is referred to as *smoothing* or *filtering* process. To be of any practical use, this process ought to employ only such information that is available from a finite approximation to the function, namely a finite number of its expansion coefficients or values at grid points.

The key idea for a filtering process in a pseudospectral grid is to replace the truncated series by a smoothed one given by

$$S_N u = \sum_{k=0}^{N} \sigma_k \cdot \hat{u}_k \cdot p_k \tag{1.75}$$

where σ_k require to be real nonnegative numbers such that $\sigma_0 = 1$ and the sequence (σ_k) is decreasing for increasing k.

The function used to determine the smoothing factors σ_k is called a *filtering function* or simply a *filter*. For our purpose we introduce a typical example, namely the exponential filter of order p:

$$\sigma_k = e^{-\alpha \cdot k^p} \quad \text{with} \quad \alpha > 0 \tag{1.76}$$

Here α is the constant free to choose. It determines the fall-off in the spectral domain, i. e. how fast the coefficients decrease.

2 The GBSSN system

The BSSNOK formulation discussed in [4], [27] and [23], hereafter referred to as BSSN for brevity, is a modification of the well known ADM formalism of general relativity with advantages in numerical simulations. The term BSSNOK refers to Thomas W. Baumgarte, Stuart L. Shapiro, Masaru Shibata, Takashi Nakamura, Kenichi Oohara and Yasufumi Kojima. The G added in front means generalized. We will discuss the modification to the traditional BSSN formulation while we derive the system.

2.1 3+1 decomposition

The Einstein equations are originally given in a 4-dimensional form. In analogy to classical dynamics we now want to prescribe initial data at one given point in time and evolve the system form this point on. Unlike in classical dynamics this is in general very difficult. We have to deal with a 4-dimensional spacetime, therefore we do not have access to an a priori global time coordinate. So we have to consider 3-dimensional spacelike hypersurfaces Σ_t labeled by $x_0 \equiv t = \text{const}$ in the spacetime. Then we choose $\Sigma_0 \equiv \Sigma$ for our initial data, i.e. on Σ the metric components and the first time derivatives of the metric are prescribed. We have pure spacelike coordinates x_i on this hypersurface.

One can show that Σ satisfies the properties of a *Cauchy surface*, i.e. it is a subset of spacetime which is intersected by every non-spacelike, inextensible curve exactly once. Therefore, initial data on this surface determines the future and past uniquely. So starting from Σ, we can calculate the metric and its first time derivatives at the other points of the spacetime. This approach we will refer to as the *Cauchy problem of general relativity*. To derive such a 3+1 decomposition, we are following [5, p. 29] and [1, p. 64].

2.1.1 Hypersurfaces in spacetime

We assume that the spacetime $\left(M, \tilde{h}_{ab}\right)$ can be foliated into a family of nonintersecting 3-dimensional spacelike hypersurfaces Σ. Those arise, at least locally, as the level surfaces of a scalar function t that can be interpreted as a global time function. Now we define the 1-form:

$$dt_a = \nabla_a t \tag{2.1}$$

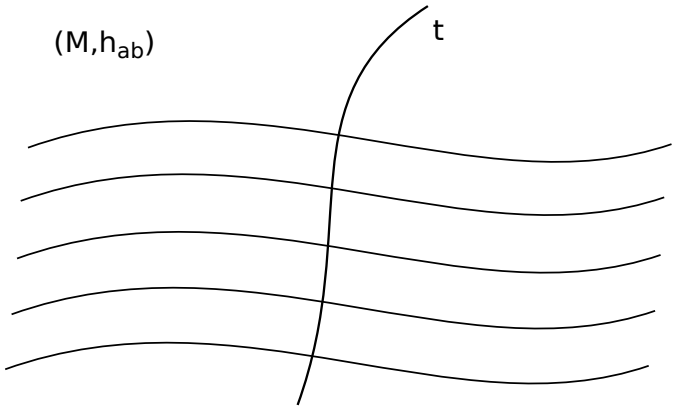


Figure 2.1: Illustration spacetime slicing.

Applying the outer derivative operator twice gives zero, hence this one-form is closed. The norm of dt can be calculated via:

$$||dt|| = \tilde{h}^{ab} dt_a dt_b =: -\alpha^{-2} \tag{2.2}$$

The function α measures how much proper time elapses between neighboring time slices along the normal vector

$$dt^a = \tilde{h}^{ab} dt_b. \tag{2.3}$$

It is called the *lapse* function. We have to assume that $\alpha > 0$, so that dt^a is timelike and the hypersurface Σ is spacelike everywhere.

Next we define the unit normal to the slices as

$$n^a := -\alpha \cdot \tilde{h}^{ab} dt_b \tag{2.4}$$

where we choose the negative sign so that n^a points in the direction of increasing t. This leads to the relation

$$n^a dt_a = -\alpha^2 \cdot \tilde{h}^{ab} dt_a dt_b = 1. \tag{2.5}$$

Furthermore, we get that

$$n^a n_a = \alpha^2 \cdot \tilde{h}^{ab} dt_a dt_b = -1. \tag{2.6}$$

So n^a is normalized and timelike and can be interpreted as the 4-velocity of an observer whose worldline is always normal to the hypersurfaces Σ.

Our next step is to construct the induced metric $\tilde{g}_{ij}$ on the spacelike hypersurface Σ. We define the tensor

$$\tilde{g}_{ab} := \tilde{h}_{ab} + n_a n_b. \tag{2.7}$$

Considering the contraction with the normal vector n^a we get

$$n^a \tilde{g}_{ab} = n^a \tilde{h}_{ab} + n^a n_a n_b = n_b - n_b = 0 \tag{2.8}$$

which shows that $\tilde{g}_{ab}$ can be viewed as a projection operator onto the spacelike hypersurface Σ (but attention: the result of the projection of a vector is a 1-form!), or alternatively: For two tangent vectors of the 4-manifold in a point of the hypersurface, their inner product is calculated by the 4-metric. From the result the inner product of their projections onto the normal vector is subtracted. So one gets only the contribution of the components lying in the hypersurface. Hence this tensor induces a 3-dimensional Riemannian metric $\tilde{g}_{ij}$ on the hypersurface Σ by the restriction to elements of the tangent space of the hypersurface. The inverse spatial metric can be found by raising the indices of the projection operator:

$$\tilde{g}^{ab} = \tilde{h}^{ac}\tilde{h}^{bd}\tilde{g}_{cd} = \tilde{h}^{ab} + n^a n^b \tag{2.9}$$

To get a more suitable definition for a projection from the tangent space of the 4-manifold to the tangent space of the hypersurface in the points of the hypersurface we raise one of the indices of $\tilde{g}_{ab}$:

$$\tilde{g}^a{}_b := \tilde{h}^a{}_b + n^a n_b = \delta^a{}_b + n^a n_b \tag{2.10}$$

With this definition we get a projection operator P which assigns to each higher rank 4-dimensional tensor its restriction to the 3-dimensional version on the hypersurface. Therefore we only have to apply it to all free indices of the tensor.

Example 2.1. Consider a vector v^b and a second rank tensor T_{cd}:

- We get $\mathrm{P}(v^b) = \tilde{g}^a{}_b v^b$ and
- $\mathrm{P}(T_{cd}) = \tilde{g}^c{}_a \tilde{g}^d{}_b T_{cd}$.

As expected the projection onto the normal vector n^a reads simply

$$N^a{}_b := -n^a n_b = \delta^a{}_b - \tilde{g}^a{}_b. \tag{2.11}$$

Example 2.2. We can decompose each vector v^a as:

$$v^a = \delta^a{}_b v^b = \left(\tilde{g}^a{}_b + N^a{}_b\right) v^b = \tilde{g}^a{}_b v^b - n^a n_b v^b \tag{2.12}$$

The next object we have to consider is the 4-dimensional covariant derivative ∇_a. We can define the covariant derivative $\tilde{\mathrm{D}}$ on the hypersurface Σ via the projection of the 4-dimensional

one. For a scalar function f, a vector v^a and a 1-1-tensor $T^a{}_b$ this reads:

$$\tilde{\mathrm{D}}_a f = \tilde{g}^{a'}{}_a \nabla_{a'} f \tag{2.13a}$$

$$\tilde{\mathrm{D}}_a v^b = \tilde{g}^{a'}{}_a \tilde{g}^b{}_{b'} \nabla_{a'} v^{b'} \tag{2.13b}$$

$$\tilde{\mathrm{D}}_a T^b{}_c = \tilde{g}^{a'}{}_a \tilde{g}^b{}_{b'} \tilde{g}^{c'}{}_c \nabla_{a'} T^{b'}{}_{c'} \tag{2.13c}$$

This induced covariant derivative can be described in local coordinates by *Christoffel symbols*. They are defined as usual by the metric $\tilde{g}_{ij}$, in this case the one on the hypersurface. This reads in the 4-dimensional representation:

$${}^{\Sigma}\Gamma^a{}_{bc} = \frac{1}{2} \cdot \tilde{g}^{ad} \left(\partial_c \tilde{g}_{db} + \partial_b \tilde{g}_{dc} - \partial_d \tilde{g}_{bc} \right) \tag{2.14}$$

Then the 3-dimensional *Riemann tensor* associated to $\tilde{g}_{ij}$ is defined by the following relations for a spatial co-vector w_d:

$$2 \cdot \tilde{\mathrm{D}}_{[a} \tilde{\mathrm{D}}_{b]} w_c = {}^{\Sigma}R^d{}_{cba} w_d \qquad\qquad {}^{\Sigma}R^d{}_{cba} n_d = 0 \tag{2.15}$$

By choosing a coordinate base we can calculate the components of the 3-dimensional Riemann tensor by the use of:

$${}^{\Sigma}R^d{}_{cba} = \partial_b {}^{\Sigma}\Gamma^d{}_{ac} - \partial_a {}^{\Sigma}\Gamma^d{}_{bc} + {}^{\Sigma}\Gamma^e{}_{ac} {}^{\Sigma}\Gamma^d{}_{eb} - {}^{\Sigma}\Gamma^e{}_{bc} {}^{\Sigma}\Gamma^d{}_{ea} \tag{2.16}$$

Of course, we get again the 3-dimensional *Ricci tensor* and *Ricci scalar* due to appropriate contractions:

$${}^{\Sigma}R_{ab} = {}^{\Sigma}R^c{}_{acb} \qquad\qquad {}^{\Sigma}R = {}^{\Sigma}R^a{}_a \tag{2.17}$$

2.1.2 Extrinsic curvature and Lie derivative

After describing 3-dimensional spacelike hypersurfaces in the 4-dimensional spacetime we will focus know on the change of these surfaces in relation to the time function. For that we define the *extrinsic curvature*:

$$\tilde{K}_{ab} = -\tilde{g}_a{}^c \tilde{g}_b{}^d \nabla_{(c} n_{d)} = -\tilde{g}_a{}^c \tilde{g}_b{}^d \nabla_c n_d \tag{2.18}$$

If there is no risk of confusion we suppress the index Σ in order to avoid cumbersome relations. The extrinsic curvature is by definition symmetric. Furthermore we call it purely spatial because it satisfies the relations

$$\tilde{K}_{ab} n^b = 0 \qquad \text{and} \qquad \tilde{K}_{ab} n^a = 0. \tag{2.19}$$

One interpretation of this tensor is that the extrinsic curvature measures how much normal vectors to the hypersurface differ at neighboring points. So in some sense it measures at

which rate the hypersurface wraps as it is carried forward along a normal vector.

Now we may rewrite the expression for the extrinsic curvature using the identity $n^d \nabla_c n_d = 0$ and get:

$$\begin{aligned} \tilde{K}_{ab} &= -\tilde{g}_a{}^c \tilde{g}_b{}^d \nabla_c n_d = -\left(\delta_a{}^c + n_a n^c\right)\left(\delta_b{}^d + n_b n^d\right)\nabla_c n_d = \\ &= -\left(\delta_a{}^c + n_a n^c\right)\delta_b{}^d \nabla_c n_d = -\nabla_a n_b - n_a n^c \nabla_c n_b \end{aligned} \tag{2.20}$$

Next we define the *Lie derivative* $\mathcal{L}_X$, following [5, p. 601]. According to the abstract definition the Lie derivative applied to a tensor measures the change of the tensor under the flow of the vector field X at the flow parameter value $\tau = 0$. The detailed development of this concept is beyond the scope of this thesis, so we content ourself with the explicit definition of the Lie derivative for simple tensors.

First of all, for a scalar function f the Lie derivative reduces to the partial derivative.

$$\mathcal{L}_X f = X^b \nabla_b f = X^b \partial_b f \tag{2.21}$$

The Lie derivative applied to a vector field v^a is the commutator

$$\mathcal{L}_X v^a = X^b \nabla_b v^a - v^b \nabla_b X^a = [X, v]^a \tag{2.22}$$

while on a one-form it reads:

$$\mathcal{L}_X w_a = X^b \nabla_b w_a + w_b \nabla_a X^b \tag{2.23}$$

This can be generalized appropriately to tensors of higher rank.

Next we refer to an exercise considered in [5, p. 601] which should give some useful insights:

Example 2.3. Let X^a and Y^a be vector fields and let $x^a(\lambda)$ denote the integral curves of X^a. If the Lie derivative $\mathcal{L}_X Y^a$ gives zero, then the vector field Y^a will connect points of equal λ in the family of integral curves $x^a(\lambda)$.

The reason is, that the vanishing of the Lie derivative is equivalent to the commutativity of the flows of the two vector fields. So following the flow of X^a for a parameter time λ and then following Y^a for a parameter time μ, is equivalent to first following Y^a for a parameter time μ and then following X^a for a parameter time λ. But this implies that the parameter values λ have to be independent of the choice of μ and therefore constant. Analogously this holds for the parameter μ, because the commutativity of the flows is symmetric under the interchanging of the vector fields X^a and Y^a.

As a direct application we choose Y^a to be a vector field tangential to the hypersurface Σ. Then it connects per definition points having the same coordinate time value t. Furthermore, we choose X^a to be the vector field αn^a with lapse function α and the normal vector n^a. If they satisfy $\mathcal{L}_{\alpha n^a} Y^a = 0$, the parallel transported vector field Y^a along the integral curves

of αn^a stays a spacelike hypersurface related to a constant coordinate time value t'.

The discussion of the previous example leads to the consideration of vanishing Lie derivatives of higher rank tensors and therefore to the definition of *Killing vector fields* described in [5, p. 602]. Lets consider the Lie derivative of the metric $\tilde{h}_{ab}$:

$$\mathcal{L}_X \tilde{h}_{ab} = X^c \nabla_c \tilde{h}_{ab} + \tilde{h}_{cb} \nabla_a X^c + \tilde{h}_{ca} \nabla_b X^c \tag{2.24}$$

Here ∇_a denotes the Levi Civita connection associated to the metric $\tilde{h}_{ab}$. Hence the first term vanishes and we get:

$$\mathcal{L}_X \tilde{h}_{ab} = \nabla_a X_c + \nabla_b X_c \tag{2.25}$$

A Killing vector field ξ is a vector field satisfying the equation:

$$\mathcal{L}_\xi \tilde{h}_{ab} = 0 \tag{2.26}$$

Hence a Killing vector field ξ^a generates an isometry of the spacetime, and a displacement along ξ^a leaves the metric invariant.

Returning to our hypersurface Σ we can now calculate the Lie derivative of $\tilde{g}_{ab}$

$$\begin{aligned} \mathcal{L}_n \tilde{g}_{ab} = \mathcal{L}_n \left(\tilde{h}_{ab} + n_a n_b \right) = 2 \cdot \nabla_{(a} n_{b)} + n_a \mathcal{L}_n n_b + n_b \mathcal{L}_n n_a = \\ = 2 \cdot \left(\nabla_{(a} n_{b)} + n_{(a} n^c \nabla_c n_{b)} \right) = -2 \cdot \tilde{K}_{ab} \end{aligned} \tag{2.27}$$

where we used Equation 2.22, Equation 2.25 and Equation 2.20 together with the symmetry of the extrinsic curvature. So we may interpret the extrinsic curvature as a measure for the change of the hypersurface Σ under the flow of the normal vector field n^a.

Additionally we can now define the *mean curvature* as the trace of the extrinsic curvature:

$$\tilde{K} := \tilde{h}^{ab} \tilde{K}_{ab} = \tilde{g}^{ab} \tilde{K}_{ab} \tag{2.28}$$

The equality comes from the fact that $\tilde{K}_{ab}$ is purely spatial.

2.1.3 Geometric equations for hypersurfaces

On a hypersurface the values of the tensor $\tilde{g}_{ij}$ and $\tilde{K}_{ab}$ are associated to each other and have to satisfy well defined relations. Before we turn to them we consider the induced covariant derivative applied to a spatial vector V^b. The contraction with the normal vector $n_b V^b$ gives zero per definition. Therefore it follows from the Leibniz rule that

$$n_q \nabla_p V^q = -V^q \nabla_p n_q. \tag{2.29}$$

So the covariant derivative on the hypersurface can be expressed as

$$\begin{aligned}\tilde{\mathrm{D}}_a V^b &= \tilde{g}_a{}^p \tilde{g}_q{}^b \nabla_p V^q = \tilde{g}_a{}^p \left(\tilde{h}_q{}^b + n_q n^b\right) \nabla_p V^q = \tilde{g}_a{}^p \nabla_p V^b - \tilde{g}_a{}^p n^b V^q \nabla_p n_q = \\ &= \tilde{g}_a{}^p \nabla_p V^b - n^b V^e \tilde{g}_a{}^p \tilde{g}_e{}^q \nabla_p n_q = \tilde{g}_a{}^p \nabla_p V^b + n^b V^e \tilde{K}_{ae}\end{aligned} \tag{2.30}$$

where we used the definition of the extrinsic curvature given by Equation 2.18.

Next we can use the definition of the Riemann tensor and this result for the 3-dimensional covariant derivative to find a relation between the Riemann tensor of the hypersurface Σ and the one of the manifold. Some straightforward calculation, which can be found in [5, p. 37], give the *Gauss equation*:

$${}^{\Sigma}R_{abcd} + \tilde{K}_{ac}\tilde{K}_{bd} - \tilde{K}_{ad}\tilde{K}_{cb} = \tilde{g}^p{}_a \tilde{g}^q{}_b \tilde{g}^r{}_c \tilde{g}^s{}_d {}^{M}R_{pqrs} \tag{2.31}$$

Similarly we consider the 3-dimensional covariant derivative of the extrinsic curvature $\tilde{\mathrm{D}}_a \tilde{K}_{bc}$ and optain, after some technical calculations, to the *Codazzi equation*:

$$\tilde{\mathrm{D}}_b \tilde{K}_{ac} - \tilde{\mathrm{D}}_a \tilde{K}_{bc} = \tilde{g}^p{}_a \tilde{g}^q{}_b \tilde{g}^r{}_c n^s {}^{M}R_{pqrs} \tag{2.32}$$

Finally we consider the Lie derivative of the extrinsic curvature in the direction of the normal vector field and get the *Ricci equation*:

$$\mathcal{L}_n \tilde{K}_{ab} = n^d n^c \tilde{g}^q{}_a \tilde{g}^r{}_b {}^{M}R_{drcq} - \frac{1}{\alpha} \tilde{\mathrm{D}}_a \tilde{\mathrm{D}}_b \alpha - \tilde{K}^c{}_b \tilde{K}_{ac} \tag{2.33}$$

We observe that the three equations introduced in this subsection are expressions for projections of the 4-dimensional Riemann tensor. The Gauss equation contains only spatial projections of the Riemann tensor, the Codazzi equation one normal projection and the Ricci equation two. Expressions with more normal projections vanish due to the symmetries of the Riemann tensor. So these three equations represent in some sense as much information as we can get.

2.1.4 The constraint equations

We saw in the previous subsection that the Gauss' equation and the Codazzi equation depend on the tensor $\tilde{g}_{ij}$, the extrinsic curvature $\tilde{K}_{ab}$ and the 3-dimensional covariant derivative $\tilde{\mathrm{D}}_a$ only. They can be interpreted as conditions which have to be satisfied to allow the embedding of a hypersurface Σ with given data $\left(\tilde{g}_{ij}, \tilde{K}_{ab}\right)$ into a spacetime manifold M with metric $\tilde{h}_{ab}$. Therefore we can derive from them our *constraint equations* which have to be satisfied by the solution at each hypersurface.

Following [5, p. 39], we eliminate the 4-dimensional Riemann tensor in the above equations

using the Einstein's equations

$$G_{ab} = {}^{M}R_{ab} - \frac{1}{2} \cdot {}^{M}R \cdot \tilde{g}_{ab} = 8\pi \cdot T_{ab}. \tag{2.34}$$

After contracting the Gauss equation (Equation 2.31) twice we get

$$\tilde{g}^{pr}\tilde{g}^{qs\,M}R_{pqrs} = {}^{\Sigma}R + \tilde{K}^2 - \tilde{K}_{ab}\tilde{K}^{ab}. \tag{2.35}$$

The left hand side of this equation can be rewritten as

$$2 \cdot n^p n^r G_{pr} = \tilde{g}^{pr}\tilde{g}^{qs\,M}R_{pqrs}. \tag{2.36}$$

Defining the *total energy density measured by a normal observer* n^a by

$$\rho := n_a n_b T^{ab}, \tag{2.37}$$

we obtain the *Hamiltonian constraint* equation:

$${}^{\Sigma}R + \tilde{K}^2 - \tilde{K}_{ab}\tilde{K}^{ab} = 16\pi \cdot \rho \tag{2.38}$$

Similarly, we contract the Codazzi equation (Equation 2.32) once and get

$$\tilde{\mathrm{D}}_b \tilde{K}_a{}^b - \tilde{\mathrm{D}}_a \tilde{K} = \tilde{g}^p{}_a \tilde{g}^{qr} n^{s\,M}R_{pqrs}. \tag{2.39}$$

The right hand side of this equation can be rewritten using the identity

$$\tilde{g}^p{}_a \tilde{g}^{qr} n^{s\,M}R_{pqrs} = -\tilde{g}^q{}_a n^s G_{qs} \tag{2.40}$$

to obtain

$$\tilde{\mathrm{D}}_b \tilde{K}_a{}^b - \tilde{\mathrm{D}}_a \tilde{K} = -\tilde{g}^q{}_a n^s G_{qs}. \tag{2.41}$$

The definition of the *momentum density measured by a normal observer* n^a

$$S_a := -\tilde{g}^q{}_a n^c T_{bc} \tag{2.42}$$

leads to the *momentum constraint* equation:

$$\tilde{\mathrm{D}}_b \tilde{K}_a{}^b - \tilde{\mathrm{D}}_a \tilde{K} = 8\pi \cdot S_a \tag{2.43}$$

2.1.5 The evolution equations

First we have to find a way to express a time derivative in this context. Following [5, p. 41] we see that the Lie derivative $\mathcal{L}_n$ used in Equation 2.33 is not a natural time derivative since

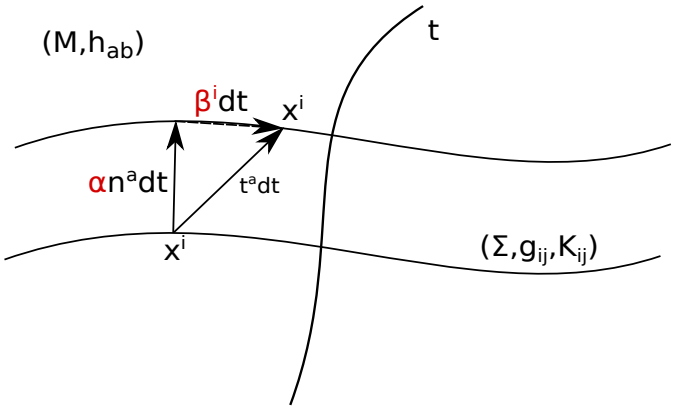


Figure 2.2: Illustration of lapse and shift.

n^a is not dual to the surface 1-form dt:

$$n^a dt_a = -\alpha \tilde{h}^{ab} \nabla_a \nabla_b t = \alpha^{-1} \tag{2.44}$$

By introducing a spatial *shift vector* β^a we can consider now the vector

$$t^a = \alpha \cdot n^a + \beta^a \tag{2.45}$$

which is dual to the 1-form dt:

$$t^a dt_a = \alpha \cdot n^a dt_a + \beta^a dt_a = 1 \tag{2.46}$$

It is useful to choose t^a to connect points with the same spatial coordinates on neighboring time slices. So we can interpret the shift vector β^a as a measure for the displacement of the spatial coordinates with respect to the normal vector n^a. Equation 2.46 implies that the integral curves of t^a are naturally parametrized by the time function t. So the flow of the vector field t^a has as flow parameter the function t and connects neighboring $t = \text{const}$ spatial hypersurfaces.

Using the linearity of the Lie derivative we get the decomposition

$$\mathcal{L}_t \tilde{K}_{ab} = \mathcal{L}_{\alpha \cdot n + \beta} \tilde{K}_{ab} = \alpha \cdot \mathcal{L}_n \tilde{K}_{ab} + \mathcal{L}_\beta \tilde{K}_{ab}. \tag{2.47}$$

By inserting the Ricci equation (Equation 2.33), following [5, p. 42], we eliminate the term $\mathcal{L}_n \tilde{K}_{ab}$ on the right hand side. Defining the *spatial stress tensor* and its trace as

$$S_{ab} := \tilde{g}^c{}_a \tilde{g}^d{}_b T_{cd} \qquad\qquad S := S^a{}_a \tag{2.48}$$

leads, after the application of the Gauss equation and Einstein's equations, to the *evolution*

equations for the extrinsic curvature:

$$\mathcal{L}_t \tilde{K}_{ab} = -\tilde{\mathrm{D}}_a \tilde{\mathrm{D}}_b \alpha + \alpha \left({}^{\Sigma}R_{ab} - 2 \cdot \tilde{K}_{ac} \tilde{K}^c{}_d + \tilde{K} \cdot \tilde{K}_{ab} \right) - 8\pi \cdot \alpha \left(S_{ab} - \frac{1}{2} \cdot \tilde{g}_{ab} \left(S - \rho \right) \right) + \mathcal{L}_\beta \tilde{K}_{ab} \tag{2.49}$$

All terms in this equation are solely associated to the 3-dimensional spatial hypersurface Σ.

From Equation 2.27 we get an expression for the extrinsic curvature as a Lie derivative of the spatial metric. Using Equation 2.47 we can rewrite this expression as

$$\mathcal{L}_t \tilde{g}_{ab} = -2\alpha \cdot \tilde{K}_{ab} + \mathcal{L}_\beta \tilde{g}_{ab} \tag{2.50}$$

which are the *evolution equations for the spatial metric.*

This coupled system of evolution equations for the extrinsic curvature and the spatial metric determine the evolution of the gravitational field data $\left(\tilde{g}_{ab}, \tilde{K}_{ab} \right)$. Together with the constraint equations (Equation 2.38 and Equation 2.43) they are completely equivalent to the Einstein equations. It can be shown that the evolution equations conserve the constraint equations, i.e. if the field data satisfies the constraint equations at some time t_0 and is evolved with the evolution equations, then it satisfies the constraint equations at all later times.

2.1.6 The standard 3+1 or ADM equations

Following [5, p. 43], we will derive the ADM equations of general relativity. For the hypersurface Σ we choose a set of spatial basis vectors $\{e^a_{(1)}, e^a_{(2)}, e^a_{(3)}\}$ which satisfy

$$dt_a e^a_{(i)} = 0. \tag{2.51}$$

They can be extended to other time slices by requiring

$$\mathcal{L}_t e^a_{(i)} = 0. \tag{2.52}$$

We choose as a fourth basis vector

$$e^a_{(0)} = t^a. \tag{2.53}$$

Together with Equation 2.46 this implies that the components of $e^a_{(0)}$ are given by

$$t^a = e^a_{(0)} = (1, 0, 0, 0), \tag{2.54}$$

so the Lie derivative along t^a reduces to a partial derivative with respect to t:

$$\mathcal{L}_t = \partial_t \tag{2.55}$$

From Equation 2.51 we get

$$0 = dt_a e^a_{(i)} = -\frac{1}{\alpha} n_a e^a_{(i)} \tag{2.56}$$

which implies the vanishing of all spatial components of n_a. From that it follows that all components of spatial tensors with a contravariant index equal to zero must vanish, because those tensors vanish when they are contracted with the normal vector. So we get:

$$\beta^a = \left(0, \beta^i\right) \tag{2.57}$$

If we solve Equation 2.45 for n^a we obtain

$$n^a = \left(\alpha^{-1}, -\alpha^{-1}\beta^i\right). \tag{2.58}$$

From the normalization condition

$$n_a n^a = -1 \tag{2.59}$$

it follows that

$$n_a = (-\alpha, 0, 0, 0). \tag{2.60}$$

Using Equation 2.7 we obtain the equality of the spatial parts of the spacetime metric $\tilde{h}_{ab}$ and the tensor $\tilde{g}_{ab}$

$$\tilde{g}_{ij} = \tilde{h}_{ij}. \tag{2.61}$$

Hence this spatial part is the Riemannian metric of the hypersurface Σ.

With the definitions made in this subsection we can now express the space time metric and its inverse as:

$$\tilde{h}_{ab} = \begin{pmatrix} -\alpha^2 + \beta_l\beta^l & \beta_i \\ \beta_j & \tilde{g}_{ij} \end{pmatrix} \qquad \tilde{h}^{ab} = \begin{pmatrix} -\alpha^{-2} & \alpha^{-2}\beta^i \\ \alpha^{-2}\beta^j & \tilde{g}^{ij} - \alpha^{-2}\beta^i\beta^j \end{pmatrix} \tag{2.62}$$

This can be written as the *standard line element in 3+1 form*:

$$ds^2 = -\alpha^2 dt^2 + \tilde{g}_{ij}\left(dx^i + \beta^i dt\right)\left(dx^j + \beta^j dt\right) \tag{2.63}$$

This allows us to rewrite our previously derived constraint and evolution system to be formulated with purely 3-dimensional objects as in [5, p. 47]. Therefore we have to take into account that any spatial tensor is already fully described by its spatial part. We saw this already for the contravariant components, but this is also true for the covariant ones. The Lie derivatives with respect to the shift vector give some additional terms with covariant derivatives. Those reduce to partial derivatives in the case of the evolution equations for the extrinsic curvature.

Finally we arrive at the *standard 3+1 or ADM equations*:

$$^{\Sigma}R + \tilde{K}^2 - \tilde{K}_{ij}\tilde{K}^{ij} = 16\pi\rho \tag{2.64a}$$

$$\tilde{\mathrm{D}}_j\left(\tilde{K}^{ij} - \tilde{g}^{ij}\tilde{K}\right) = 8\pi S^i \tag{2.64b}$$

$$\partial_t \tilde{g}_{ij} = -2\alpha \cdot \tilde{K}_{ij} + \tilde{\mathrm{D}}_i \beta_j + \tilde{\mathrm{D}}_j \beta_i \tag{2.64c}$$

$$\begin{aligned}\partial_t \tilde{K}_{ij} = {} & \alpha \left({}^{\Sigma}R_{ij} - 2\tilde{K}_{ik}\tilde{K}^k{}_j + \tilde{K} \cdot \tilde{K}_{ij} \right) - \tilde{\mathrm{D}}_i \tilde{\mathrm{D}}_j \, \alpha - 8\pi\alpha \left(S_{ij} - \frac{1}{2}\tilde{g}_{ij} \left(S - \rho \right) \right) \\ & + \beta^k \partial_k \tilde{K}_{ij} + \tilde{K}_{ik} \partial_j \beta^k + \tilde{K}_{kj} \partial_i \beta^k \end{aligned} \tag{2.64d}$$

2.2 Derivation of the generalized BSSN system

In this section we will use the results of the previous one to derive the GBSSN system. This derivation involves a large amount of straightforward calculations, which we will skip largely. Instead we will focus on the ideas behind. A more detailed approach can be found in [9] and [8] which also include a lot of references to further details.

First we consider the derivative ∂_t defined in Equation 2.55 as the Lie derivative in the direction of the vector field t^a. For the sake of simplicity we use the notation commonly used in literature, i.e. we define the derivative operator

$$\partial_\perp := \partial_t - \mathcal{L}_\beta = \alpha \cdot \mathcal{L}_n, \tag{2.65}$$

where the last equality follows from Equation 2.47 and Equation 2.55.

Next we restrict our considerations to the vacuum case. This is what we will use in the simulations later on. Then Equation 2.64c and Equation 2.64d give:

$$\partial_\perp \tilde{g}_{ij} = -2\alpha \cdot \tilde{K}_{ij} \tag{2.66a}$$

$$\partial_\perp \tilde{K}_{ij} = \alpha \tilde{K} \tilde{K}_{ij} - 2\alpha \tilde{K}_{ik} \tilde{K}^k{}_j + \alpha \, {}^{\Sigma}R_{ij} - \tilde{\mathrm{D}}_i \tilde{\mathrm{D}}_j \, \alpha \tag{2.66b}$$

Additionally the constraints given by Equation 2.64a and Equation 2.64b turn into:

$$\begin{aligned} \mathcal{H} &= \tilde{K}^2 - \tilde{K}_{ij}\tilde{K}^{ij} + {}^{\Sigma}R = 0 \\ \mathcal{M}_i &= \tilde{\mathrm{D}}_j \, \tilde{K}^j{}_i - \tilde{\mathrm{D}}_j \, \tilde{K} = 0 \end{aligned} \tag{2.67}$$

To arrive at the GBSSN variables we have to apply a conformal transformation to the spatial metric and the extrinsic curvature:

$$\tilde{g}_{ij} = e^{4\phi} g_{ij} \tag{2.68a}$$

$$\tilde{K}_{ij} = e^{4\phi} \cdot K_{ij} = e^{4\phi} \left(A_{ij} + \frac{1}{3} g_{ij} K \right) \tag{2.68b}$$

Here A_{ij} and K denote the trace-free part and the trace of K_{ij}. The choice of $e^{4\phi}$ as conformal factor is a matter of convenience and satisfies

$$e^{4\phi} = \left(\frac{\tilde{g}}{g} \right)^{\frac{1}{3}} \tag{2.69}$$

where $\tilde{g}$ and g denote the determinants of the spatial and conformal metric respectively. Then these defining relations can be inverted purely algebraically to get:

$$\phi = \frac{1}{12} \ln \left(\frac{\tilde{g}}{g} \right) \tag{2.70a}$$

$$g_{ij} = \left(\frac{\tilde{g}}{g} \right)^{-\frac{1}{3}} \tilde{g}_{ij} \tag{2.70b}$$

$$A_{ij} = \left(\frac{\tilde{g}}{g} \right)^{-\frac{1}{3}} \left(\tilde{K}_{ij} - \frac{1}{3} \tilde{g}_{ij} \left(\tilde{K} - A \right) \right) \tag{2.70c}$$

$$K = \tilde{K} - A \tag{2.70d}$$

Here the terms K, $\tilde{K}$ and A denote the conformal and physical mean curvature and the so to say trace of the trace-free part. Of course A will be zero, but we will keep it for the moment. The left hand side of this system shows already some the variables of the GBSSN system. We require evolution equations, hence we calculate their $\partial_\perp$ derivatives. Using identities of the conformal spatial Ricci tensor R_{ij} and the conformal spatial covariant derivative $\tilde{\mathrm{D}}$

$${}^{\Sigma}R_{ij} = R_{ij} - 2\,\mathrm{D}_i\,\mathrm{D}_j\,\phi + 4\,\mathrm{D}_i\,\phi\,\mathrm{D}_j\,\phi - 2g_{ij}\left(\mathrm{D}^2\,\phi + 2\,\mathrm{D}^k\,\phi\,\mathrm{D}_k\,\phi\right) \tag{2.71a}$$

$$\tilde{\mathrm{D}}_i\,\tilde{\mathrm{D}}_j\,\alpha = \mathrm{D}_i\,\mathrm{D}_j\,\alpha - 4\,\mathrm{D}_{(i}\,\alpha\,\mathrm{D}_{j)}\,\phi + 2g_{ij}\,\mathrm{D}^k\,\alpha\,\mathrm{D}_k\,\phi \tag{2.71b}$$

leads to:

$$\partial_\perp \phi = -\frac{1}{12}\partial_\perp \ln g - \frac{1}{6}\alpha\,(K + A) \tag{2.72a}$$

$$\partial_\perp g_{ij} = \frac{1}{3} g_{ij}\partial_\perp \ln g - 2\alpha A_{ij} + \frac{2}{3}\alpha g_{ij} A \tag{2.72b}$$

$$\begin{aligned}\partial_\perp A_{ij} = {} & \frac{1}{3} A_{ij}\partial_\perp \ln g + \frac{1}{3} g_{ij}\partial_\perp A - 2\alpha A_{ik}A^k{}_j + \alpha A_{ij}K + \frac{1}{3}\alpha A\left(5A_{ij} - Ag_{ij} - Kg_{ij}\right) \\ & + \frac{1}{e^{4\phi}}\left(-2\alpha\,\mathrm{D}_i\,\mathrm{D}_j\,\phi + 4\alpha\,\mathrm{D}_i\,\phi\,\mathrm{D}_j\,\phi + 4\,\mathrm{D}_{(i}\,\alpha\,\mathrm{D}_{j)}\,\phi - \mathrm{D}_i\,\mathrm{D}_j\,\alpha + \alpha R_{ij}\right)^{\mathrm{TF}}\end{aligned} \tag{2.72c}$$

$$\partial_\perp K = -\partial_\perp A + \alpha\,(K + A)^2 + \frac{1}{e^{4\phi}}\left(\alpha R - 8\alpha\,\mathrm{D}^i\,\phi\,\mathrm{D}_i\,\phi - 8\alpha\,\mathrm{D}^2\,\phi - \mathrm{D}^2\,\alpha - 2\,\mathrm{D}^i\,\alpha\,\mathrm{D}_i\,\phi\right) \tag{2.72d}$$

Here D^2 is just the Laplacian and the superscript 'TF' denotes the trace-free part of the expression. Next we define

$$\Delta\Gamma^i := g^{jk}\left(\Gamma_j{}^i{}_k - \bar{\Gamma}_j{}^i{}_k\right) \tag{2.73}$$

as the contraction of the difference between the Christoffel symbols of the conformal metric g_{ij} and the time independent Christoffel symbols of an arbitrary background metric f_{ij}. These background Christoffel symbols allow the definition of a background covariant derivative $\bar{D}_i$ and a background Riemann tensor $\bar{R}^i{}_{jkl}$. Then our previously derived equation system

implies:

$$\begin{aligned}\partial_\perp\left(\Delta\Gamma^i\right) = {} & g^{jk}\bar{D}_j\bar{D}_k\beta^i - g^{jk}\bar{R}^i{}_{jkl}\beta^l - 2A^{ij}\partial_j\alpha - \frac{2\alpha}{\sqrt{g}}\bar{D}_j\left(\sqrt{g}A^{ij}\right) \\ & - \frac{1}{3}\Delta\Gamma^i\left(\partial_\perp \ln g - 4\alpha A\right) - \frac{1}{6}g^{ij}\partial_j\left(\partial_\perp \ln g - 4\alpha A\right)\end{aligned} \tag{2.74}$$

This allows us to define a new GBSSN variable Λ^i, which is equal to $\Delta\Gamma^i$. The right hand side of the derivative of this new variable is also equal to the expression in the previous equation. So we get an additional constraint C^i and the constraint system can be written as:

$$\mathcal{H} = \frac{2}{3}\left(K + A\right)^2 + \frac{1}{3}A^2 - A_{ij}A^{ij} + \frac{1}{e^{4\phi}}\left(R - 8\,\mathrm{D}^i\,\phi\,\mathrm{D}_i\,\phi - 8\,\mathrm{D}^2\,\phi\right) \tag{2.75a}$$

$$\begin{aligned}\mathcal{M}^i = {} & \frac{1}{\sqrt{g}\cdot e^{4\phi}}\bar{D}_j\left(\sqrt{g}A^{ij}\right) + 6\frac{1}{e^{4\phi}}\left(A^{ij} - \frac{1}{3}Ag^{ij}\right)\partial_j\phi \\ & - \frac{1}{e^{4\phi}}g^{ij}\partial_j\left(\frac{2}{3}K + A\right) + \frac{1}{e^{4\phi}}A^{jk}\left(\Gamma_j{}^i{}_k - \bar{\Gamma}_j{}^i{}_k\right)\end{aligned} \tag{2.75b}$$

$$C^i = \Lambda^i - \Delta\Gamma^i = 0 \tag{2.75c}$$

Next we add $-\alpha\mathcal{H}$ to the right hand side of $\partial_\perp K$ and $2\alpha e^{4\phi}\mathcal{M}^i$ to right hand side of $\partial_\perp\left(\Lambda^i\right)$. Then we have to simplify the equations. We now set $A = 0$ and $\partial_\perp A = 0$. Some technical calculations, which we skip here, lead to the GBSSN system, in this form also found in [8, p. 2]:

$$\partial_\perp\phi = -\frac{1}{12}\partial_\perp \ln g - \frac{1}{6}\alpha K \tag{2.76a}$$

$$\partial_\perp g_{ij} = \frac{1}{3}g_{ij}\partial_\perp \ln g - 2\alpha A_{ij} \tag{2.76b}$$

$$\partial_\perp A_{ij} = \frac{1}{3}A_{ij}\partial_\perp \ln g - 2\alpha A_{ik}A^k{}_j + \alpha A_{ij}K + \frac{1}{e^{4\phi}}\left(\alpha^\Sigma R_{ij} - \tilde{\mathrm{D}}_i\,\tilde{\mathrm{D}}_j\,\alpha\right)^{\mathrm{TF}} \tag{2.76c}$$

$$\partial_\perp K = \frac{1}{3}\alpha K^2 + \alpha A_{ij}A^{ij} - \tilde{\mathrm{D}}^i\,\tilde{\mathrm{D}}_i\,\alpha \tag{2.76d}$$

$$\partial_\perp\Lambda^i = -\frac{1}{3}\Lambda^i\partial_\perp \ln g - \frac{1}{6}g^{ij}\partial_j\partial_\perp \ln g - 2A^{ij}\partial_j\alpha + 2\alpha\left(\Gamma_j{}^i{}_k A^{jk} + 6A^{ij}\partial_j\phi - \frac{2}{3}g^{ij}\partial_j K\right) \tag{2.76e}$$

At this point we will discuss the difference between the original BSSN system and the generalized one. In the original one the determinant of the conformal metric was fixed to $g = 1$. This causes problems if we want to consider spherical symmetry. In the GBSSN system we are not restricted to this choice anymore and we are free to let g evolve in time.

There are two natural choices as discussed in [9] and [8]:

$$\partial_\perp \ln g = 0 \qquad \text{(Eulerian condition)} \tag{2.77a}$$

$$\partial_t \ln g = 0 \qquad \text{(Lagrangian condition)} \tag{2.77b}$$

The Lagrangian condition implies $\partial_\perp \ln g = -2\,\mathrm{D}_i\,\beta^j$.

In the original formulation the expression $g = 1$ appeared as an additional constraint. There is no clear equivalence between GBSSN and BSSN. In GBSSN 16 variables are evolved (ϕ, 6 components of g_{ij}, 5 components of A_{ij}, K and the 3 components of Λ^i), in BSSN only 15. Nevertheless simulations show that the GBSSN system with the Lagrangian condition behaves very similar to the traditional BSSN system as shown and discussed in [8, p. 2]. Therefore we choose for our simulations the Lagrangian condition $\partial_\perp \ln g = -2\,\mathrm{D}_i\,\beta^j$.

2.3 GBSSN in spherical symmetry

In this section we will derive the explicit equations used in the simulations later on. First we state the general form of a 3-dimensional spatial spherical symmetric line element:

$$ds^2 = A(t,r)dr^2 + B(t,r)r^2\left(d\theta^2 + \sin^2(\theta)d\phi^2\right) \tag{2.78}$$

Following [8, p. 3] we now make an ansatz for the conformal metric:

$$g_{ij} = \begin{pmatrix} g_{rr} & 0 & 0 \\ 0 & r^2 \cdot g_{\theta\theta} & 0 \\ 0 & 0 & r^2 \cdot g_{\theta\theta} \cdot \sin^2\theta \end{pmatrix} \tag{2.79}$$

Note that in comparison to [8, p. 3] a factor of r^2 is pulled out from $g_{\theta\theta}$. The ansatz for the trace-free part of the extrinsic curvature is given by:

$$A_{ij} = A_{rr} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & -\frac{r^2 \cdot g_{\theta\theta}}{2g_{rr}} & 0 \\ 0 & 0 & -\frac{r^2 \cdot g_{\theta\theta} \cdot \sin^2\theta}{2g_{rr}} \end{pmatrix} \tag{2.80}$$

Additionally we can, due to symmetry properties, use the following ansatz for the variable Λ^i:

$$\Lambda^i = \begin{pmatrix} \Lambda^r \\ -\frac{\cos\theta}{r^2 \cdot g_{\theta\theta} \sin\theta} \\ 0 \end{pmatrix} \tag{2.81}$$

For the sake of convenience we replace the variable ϕ by $\chi := \frac{1}{e^{4\phi}}$. Let β^r denote the radial shift component, then the GBSSN line element reads:

$$ds^2 = -\left(\alpha^2 - \frac{(\beta^r)^2 g_{rr}}{\chi}\right) dt^2 + \frac{2\beta^r g_{rr}}{\chi} dt\, dr + \frac{g_{rr}}{\chi} dr^2 + \frac{g_{\theta\theta}}{\chi} r^2 \left(d\theta^2 + \sin^2\theta \cdot d\phi\right) \tag{2.82}$$

Next we apply the introduced ansatz and switch back to the time derivative ∂_t. After some calculations we get the following explicit *GBSSN formulation in spherical symmetry* which we will evolve during our simulations:

$$\partial_t \chi = \frac{2\alpha\chi K}{3} - \frac{2}{3}\chi\nu(\beta^r)' + \beta^r\chi' - \frac{\beta^r\chi\nu g'_{rr}}{3g_{rr}} - \frac{2\beta^r\chi\nu g'_{\theta\theta}}{3g_{\theta\theta}} - \frac{4\beta^r\chi\nu}{3r} \tag{2.83a}$$

$$\partial_t g_{rr} = -2\alpha A_{rr} - \frac{2}{3} g_{rr}\nu(\beta^r)' + 2g_{rr}(\beta^r)' - \frac{1}{3}\beta^r\nu g'_{rr} + \beta^r g'_{rr} - \frac{2\beta^r g_{rr}\nu g'_{\theta\theta}}{3g_{\theta\theta}} - \frac{4\beta^r g_{rr}\nu}{3r} \tag{2.83b}$$

$$\partial_t g_{\theta\theta} = \frac{\alpha A_{rr} g_{\theta\theta}}{g_{rr}} - \frac{2}{3} g_{\theta\theta}\nu(\beta^r)' - \frac{\beta^r g_{\theta\theta}\nu g'_{rr}}{3g_{rr}} - \frac{2}{3}\beta^r\nu g'_{\theta\theta} + \beta^r g'_{\theta\theta} - \frac{4\beta^r g_{\theta\theta}\nu}{3r} + \frac{2\beta^r g_{\theta\theta}}{r} \tag{2.83c}$$

$$\begin{aligned}
\partial_t A_{rr} = &-\frac{2\chi\alpha''}{3} - \frac{2\alpha'\chi'}{3} + \frac{\chi\alpha' g'_{rr}}{3g_{rr}} + \frac{\chi\alpha' g'_{\theta\theta}}{3g_{\theta\theta}} + \frac{2\chi\alpha'}{3r} - \frac{2\alpha A_{rr}^2}{g_{rr}} + \alpha A_{rr} K + \frac{\alpha\chi''}{3} - \frac{\alpha\chi' g'_{rr}}{6g_{rr}} \\
&- \frac{\alpha\chi' g'_{\theta\theta}}{6g_{\theta\theta}} - \frac{\alpha\chi'}{3r} - \frac{\alpha\left(\chi'\right)^2}{6\chi} - \frac{\alpha\chi g''_{rr}}{3g_{rr}} - \frac{\alpha\chi g'_{rr} g'_{rr}}{2g_{rr} g_{rrb}} - \frac{\alpha\chi g'_{rr} g'_{\theta\theta}}{2g_{rr} g_{\theta\theta}} + \frac{\alpha\chi g'_{rr} g'_{\theta\theta}}{6g_{rr} g_{\theta\theta b}} - \frac{2\alpha\chi g'_{rr}}{3g_{rr} r} \\
&+ \frac{2\alpha\chi\left(g'_{rr}\right)^2}{3g_{rr}^2} + \frac{\alpha\chi g_{rr}\Lambda^r g'_{rr}}{3g_{rrb}} + \frac{\alpha\chi g_{rr} g'_{rr} g'_{\theta\theta}}{3g_{rrb}^2 g_{\theta\theta}} + \frac{2\alpha\chi g_{rr} g_{\theta\theta b} g'_{rr}}{3g_{rrb}^2 g_{\theta\theta} r} + \frac{2\alpha\chi g_{rr} g'_{\theta\theta} g'_{\theta\theta}}{3g_{rrb} g_{\theta\theta}^2} \\
&+ \frac{4\alpha\chi g_{rr} g_{\theta\theta b} g'_{\theta\theta}}{3g_{rrb} g_{\theta\theta}^2 r} - \frac{2\alpha\chi g_{rr} g''_{\theta\theta}}{3g_{rrb} g_{\theta\theta}} + \frac{\alpha\chi g_{rr}\left(g'_{\theta\theta}\right)^2}{3g_{rrb} g_{\theta\theta} g_{\theta\theta b}} + \frac{8\alpha\chi g_{rr} g_{\theta\theta b}}{3g_{rrb} g_{\theta\theta} r^2} - \frac{2\alpha\chi g_{rr}}{3g_{\theta\theta} r^2} - \frac{\alpha\chi g_{rr}\Lambda^r g'_{\theta\theta}}{3g_{\theta\theta b}} \\
&+ \frac{2}{3}\alpha\chi g_{rr}(\Lambda^r)' - \frac{2\alpha\chi g_{rr}\Lambda^r}{3r} + \frac{\alpha\chi g''_{rr}}{3g_{rrb}} + \frac{\alpha\chi g'_{rr} g'_{\theta\theta}}{3g_{rrb} g_{\theta\theta}} - \frac{\alpha\chi g'_{rr} g'_{\theta\theta}}{6g_{rrb} g_{\theta\theta b}} + \frac{\alpha\chi g'_{rr}}{3g_{rrb} r} - \frac{\alpha\chi\left(g'_{rr}\right)^2}{6g_{rrb}^2} \\
&+ \frac{\alpha\chi g''_{\theta\theta}}{3g_{\theta\theta}} - \frac{\alpha\chi g'_{\theta\theta} g'_{\theta\theta}}{3g_{\theta\theta} g_{\theta\theta b}} - \frac{2\alpha\chi g'_{\theta\theta}}{3g_{\theta\theta} r} - \frac{\alpha\chi\left(g'_{\theta\theta}\right)^2}{3g_{\theta\theta}^2} - \frac{2\alpha\chi g'_{\theta\theta}}{3g_{\theta\theta b} r} - \frac{2\alpha\chi}{r^2} + \beta^r A'_{rr} - \frac{2}{3} A_{rr}\nu(\beta^r)' \\
&+ 2A_{rr}(\beta^r)' - \frac{A_{rr}\beta^r\nu g'_{rr}}{3g_{rr}} - \frac{2A_{rr}\beta^r\nu g'_{\theta\theta}}{3g_{\theta\theta}} - \frac{4A_{rr}\beta^r\nu}{3r}
\end{aligned} \tag{2.83d}$$

$$\partial_t K = -\frac{\chi\alpha''}{g_{rr}} + \frac{\alpha'\chi'}{2g_{rr}} + \frac{\chi\alpha' g'_{rr}}{2g_{rr}^2} - \frac{\chi\alpha' g'_{\theta\theta}}{g_{rr} g_{\theta\theta}} - \frac{2\chi\alpha'}{g_{rr} r} + \frac{3\alpha A_{rr}^2}{2g_{rr}^2} + \frac{\alpha K^2}{3} + \beta^r K' \tag{2.83e}$$

$$\begin{aligned}
\partial_t \Lambda^r = &-\frac{2A_{rr}\alpha'}{g_{rr}^2} - \frac{3\alpha A_{rr}\chi'}{\chi g_{rr}^2} - \frac{\alpha A_{rr} g'_{rr}}{g_{rr}^2 g_{rrb}} + \frac{\alpha A_{rr} g'_{\theta\theta}}{g_{rr}^2 g_{\theta\theta}} + \frac{2\alpha A_{rr}}{g_{rr}^2 r} + \frac{\alpha A_{rr} g'_{rr}}{g_{rr}^3} \\
&- \frac{\alpha A_{rr} g'_{\theta\theta}}{g_{rr} g_{rrb} g_{\theta\theta}} - \frac{2\alpha A_{rr} g_{\theta\theta b}}{g_{rr} g_{rrb} g_{\theta\theta} r} - \frac{4\alpha K'}{3g_{rr}} + \frac{\nu(\beta^r)''}{3g_{rr}} + \frac{(\beta^r)''}{g_{rr}} + \frac{\nu(\beta^r)' g'_{rr}}{2g_{rr}^2} - \frac{(\beta^r)' g'_{rr}}{2g_{rr}^2} \\
&- \frac{\nu(\beta^r)' g'_{rr}}{3g_{rr} g_{rrb}} + \frac{(\beta^r)' g'_{rr}}{g_{rr} g_{rrb}} - \frac{\nu(\beta^r)' g'_{\theta\theta}}{3g_{rr} g_{\theta\theta}} + \frac{(\beta^r)' g'_{\theta\theta}}{g_{rr} g_{\theta\theta}} - \frac{2\nu(\beta^r)'}{3g_{rr} r} + \frac{2(\beta^r)'}{g_{rr} r} + \frac{2\nu(\beta^r)' g'_{\theta\theta}}{3g_{rrb} g_{\theta\theta}} \\
&+ \frac{4g_{\theta\theta b}\nu(\beta^r)'}{3g_{rrb} g_{\theta\theta} r} + \frac{\beta^r\nu g''_{rr}}{6g_{rr}^2} + \frac{\beta^r\nu g'_{rr} g'_{\theta\theta}}{3g_{rr} g_{rrb} g_{\theta\theta}} + \frac{2\beta^r g_{\theta\theta b}\nu g'_{rr}}{3g_{rr} g_{rrb} g_{\theta\theta} r} - \frac{\beta^r\nu g'_{rr} g'_{rr}}{6g_{rr}^2 g_{rrb}} + \frac{\beta^r g''_{rr}}{2g_{rr} g_{rrb}}
\end{aligned}$$

$$
\begin{aligned}
&- \frac{\beta^r \nu g'_{rr} g'_{\theta\theta}}{3 g_{rr} g_{rrb} g_{\theta\theta}} - \frac{2\beta^r \nu g'_{rr}}{3 g_{rr} g_{rrb} r} - \frac{\beta^r \left(g'_{rr}\right)^2}{2 g_{rr} g_{rrb}^2} + \frac{\beta^r \nu g''_{\theta\theta}}{3 g_{rr} g_{\theta\theta}} - \frac{8\beta^r \nu g'_{\theta\theta}}{3 g_{rr} g_{\theta\theta} r} - \frac{\beta^r \nu \left(g'_{\theta\theta}\right)^2}{g_{rr} g_{\theta\theta}^2} - \frac{10\beta^r \nu}{3 g_{rr} r^2} \\
&+ \frac{\beta^r g'_{rr} g'_{\theta\theta}}{g_{rrb}^2 g_{\theta\theta}} + \frac{2\beta^r g_{\theta\theta b} g'_{rr}}{g_{rrb}^2 g_{\theta\theta} r} + \frac{2\beta^r \nu g'_{\theta\theta} g'_{\theta\theta}}{3 g_{rrb} g_{\theta\theta}^2} + \frac{4\beta^r g_{\theta\theta b} \nu g'_{\theta\theta}}{3 g_{rrb} g_{\theta\theta}^2 r} - \frac{\beta^r g''_{\theta\theta}}{g_{rrb} g_{\theta\theta}} + \frac{4\beta^r \nu g'_{\theta\theta}}{3 g_{rrb} g_{\theta\theta} r} \\
&- \frac{4\beta^r g'_{\theta\theta}}{g_{rrb} g_{\theta\theta} r} + \frac{8\beta^r g_{\theta\theta b} \nu}{3 g_{rrb} g_{\theta\theta} r^2} - \frac{2\beta^r g_{\theta\theta b}}{g_{rrb} g_{\theta\theta} r^2} + \beta^r (\Lambda^r)' \qquad (2.83f)
\end{aligned}
$$

Here the superscript $'$ denotes a partial derivative with respect to the coordinate variable r. The variables with additional index b refer to the background metric. The variable ν allows a selection of the Eulerian $\nu = 0$ and the Lagrangian condition $\nu = 1$.

2.4 Fixing lapse and shift

Finally, we observe that the system in Equation 2.83 requires a choice of the lapse and shift variables. To provide a little more insight, we outline a discussion in [30, p. 437].

Let (M, g) and (N, h) denote smooth pseudo-Riemannian manifolds, further denoted as M and N. A smooth, bijective mapping ϕ between M and N is called *diffeomorphism*, if its inverse is smooth too. This already implies that the dimensions of the manifolds coincide.

We can define the tangent mapping $T\phi$ of a diffeomorphism ϕ for each point $p \in M$ as a mapping from the tangent space T_pM to $T_{\phi(p)}N$ via

$$
\left(T_p\phi(v)\right)(f) := v(f \circ \phi) \qquad v \in T_pM, \quad f \in C^\infty(N, \mathbb{R}). \qquad (2.84)
$$

The same is true for the inverse of ϕ. This definition generalizes appropriately for tensors of higher order.

Alternatively stated, if $\phi : M \to N$ is a diffeomorphism, then M and N have identical manifold structure. If a theory describes nature in terms of a spacetime manifold M and tensor fields $t^{(i)}$ defined on M, where (i) denotes any number of co- or contravariant indices, then the solutions $(M, t^{(i)})$ and $(N, (T\phi(t^{(i)}))$ have physical identical properties.

However, if we are given a coordinate system $\{x^\mu\}$ covering a neighborhood U of p and a coordinate system $\{y^\mu\}$ covering a neighborhood V of $\phi(p)$, we may take the following passive point of view. We may use ϕ to define a new coordinate system $\{\bar{x}^\mu\}$ in the neighborhood $O := \phi^{-1}(V)$ of p by setting

$$
\bar{x}^\mu(q) = y^\mu(\phi(q)) \qquad \text{for } q \in O. \qquad (2.85)
$$

We then may view the effect of ϕ as leaving p and all tensors at p unchanged, but inducing the coordinate transformation $x^\mu \to \bar{x}^\mu$.

In general relativity the equation $G_{ab} = 0$ is an underdetermined system of equations for the metric components $\tilde{h}_{ab}$, because we have only 6 evolution equations for 10 unknown metric components. However, this underdetermination is not physical. It results from the

redundancy in the description.

For our purpose this leads to a freedom of choice in the coordinates, which manifests in our situation in the freedom of choice of the lapse function α and the three components of the shift vector field β. We refer to this process as *gauge fixing*.

2.4.1 Lapse

There are several common possibilities to choose the lapse function. The first one is the lapse which allows *maximal slicing*, i.e. the mean curvature and its derivative on all slices vanish

$$\tilde{K} = 0 = \partial_t \tilde{K}. \tag{2.86}$$

For a direct implementation of this condition one has to solve an elliptic equation in each time step. This leads to additional complications and higher computational costs.

The second is the *1+log slicing* condition or more precisely the family of 1+log slicing conditions. The easiest example is

$$\partial_t \alpha = -n\alpha K \tag{2.87}$$

with n a constant usually chosen as $n = 2$. In case of initial data with $K = 0$ the lapse will not evolve. Thus in case of the trumpet simulations (see Section 6.2), this trick will allow us to maintain maximal slicing and therefore a stationary solution.

A more general 1+log condition reads

$$\partial_t \alpha = \beta^i \partial_i \alpha - n\alpha K \tag{2.88}$$

which comes from our previous normal derivative $\partial_\perp \alpha = -n\alpha K$. In spherical symmetry this gives for $n = 2$

$$\partial_t \alpha = \beta^r \alpha' - 2\alpha K \tag{2.89}$$

and is our default equation for the lapse used for puncture evolution.

2.4.2 Shift

The easiest choice for the shift is the *zero shift* condition

$$\beta^i = 0. \tag{2.90}$$

Naturally this is not the most interesting case. To get stable results in moving puncture evolution of binary black holes, the usual choice for the shift is an element of the family of *Gamma-driver shift* conditions. Here the term Gamma refers to the variable $\Delta\Gamma^i$ in the previous derivation of the system which plays a role in the evolution of the shift variable. A

general version of such a shift condition is given by [5, p. 120]:

$$\partial_t \beta^i = \frac{3}{4} B^i + \beta^k \partial_k \beta^i \tag{2.91a}$$

$$\partial_t B^i = \partial_t \Delta\Gamma^i + \beta^k \partial_k B^i - \beta^k \partial_k \Delta\Gamma^i - \eta B^i \tag{2.91b}$$

Here B^i is a vector field introduced to define the shift and η is typically a factor of order $\approx \frac{2}{M}$ where M denotes the mass of a black hole in puncture evolution. The advantage of this shift condition is that it can be evolved together with the GBSSN equations and the whole system including the gauge fields α, β^i and B^i becomes hyperbolic. To get other elements of this shift condition family one can for example omit some of the advection terms on the right hand side. A typical example which is used in the puncture simulations is motivated by satisfying results in the binary black hole evolution and gives in spherical symmetry the expressions

$$\partial_t \beta^r = \frac{(\beta^r)^2 g'_{rr}}{2 g_{rrb}} + \beta^r (\beta^r)' + \frac{3 B_r}{4} \tag{2.92a}$$

$$\partial_t B_r = \beta^r B'_r + \frac{\beta^r B_r g'_{rr}}{2 g_{rrb}} - \frac{3 \beta^r \Lambda^r g'_{rr}}{8 g_{rrb}} - \frac{3 \beta^r (\Lambda^r)'}{4} - B_r \eta + \frac{3 \partial_t \Lambda^r}{4} \tag{2.92b}$$

which are also used in [16]. We will add them to our evolution system for puncture evolution. The only change in case of trumpet evolution is that we drop one term on the right hand side of the $\partial_t \beta^r$ equation:

$$\partial_t \beta^r = \frac{(\beta^r)^2 g'_{rr}}{2 g_{rrb}} + \frac{3 B_r}{4} \tag{2.93}$$

2.5 Constraints

Finally, we also state the constraint equations formulated in the variables of the GBSSN system in spherical symmetry.

$$\begin{aligned} \mathcal{H} = &- \frac{3 A_{rr}^2}{2 g_{rr}^2} + \frac{2 \chi''}{g_{rr}} - \frac{\chi' g'_{rr}}{g_{rr}^2} + \frac{2 \chi' g'_{\theta\theta}}{g_{rr} g_{\theta\theta}} + \frac{4 \chi'}{g_{rr} r} - \frac{5 (\chi')^2}{2 \chi g_{rr}} + \frac{\chi g'_{rr} g'_{\theta\theta}}{g_{rr}^2 g_{\theta\theta}} + \frac{2 \chi g'_{rr}}{g_{rr}^2 r} - \frac{2 \chi g''_{\theta\theta}}{g_{rr} g_{\theta\theta}} \\ &- \frac{6 \chi g'_{\theta\theta}}{g_{rr} g_{\theta\theta} r} + \frac{\chi (g'_{\theta\theta})^2}{2 g_{rr} g_{\theta\theta}^2} - \frac{2 \chi}{g_{rr} r^2} + \frac{2 \chi}{g_{\theta\theta} r^2} + \frac{2 K^2}{3} \end{aligned} \tag{2.94a}$$

$$\mathcal{M} = \frac{A'_{rr}}{g_{rr}} - \frac{3 A_{rr} \chi'}{2 \chi g_{rr}} - \frac{A_{rr} g'_{rr}}{g_{rr}^2} + \frac{3 A_{rr} g'_{\theta\theta}}{2 g_{rr} g_{\theta\theta}} + \frac{3 A_{rr}}{g_{rr} r} - \frac{2 K'}{3} \tag{2.94b}$$

$$C^r = - \frac{g'_{rr}}{2 g_{rr}^2} + \frac{g'_{rr}}{2 g_{rr} g_{rrb}} + \frac{g'_{\theta\theta}}{g_{rr} g_{\theta\theta}} + \frac{2}{g_{rr} r} - \frac{g'_{\theta\theta}}{g_{rrb} g_{\theta\theta}} - \frac{2 g_{\theta\theta b}}{g_{rrb} g_{\theta\theta} r} + \Lambda^r \tag{2.94c}$$

3 Systems of partial differential equations

The focus of this chapter lies on the systems of partial differential equations (PDE) for the simulations. Most of the effort is devoted to the calculation of characteristic eigenfields from the principal parts of the systems. These fields are used later on for the formulation of interface conditions in the hybrid grid and boundary conditions in general.

3.1 Hyperbolic systems

A special type of partial differential equations are the *hyperbolic* ones. The prototypical example is the wave equation. Following [5, p. 376] we introduce some definitions for hyperbolic systems and PDEs in general. We assume a time coordinate and a single space coordinate.

Then hyperbolic systems can always be recast into the following first order form by the introduction of additional variables

$$\partial_t u + A(u) \cdot \partial_x u = S(u) \tag{3.1}$$

where $S(u)$ denotes terms of lower order, i.e. without derivatives in this case.

If we can find some norm $||.||$, such that the solution of a system satisfies

$$||u(t, x^i)|| \leq k e^{\alpha t} ||u(0, x^i)|| \tag{3.2}$$

for all times t, where k and α are constants independent of the initial data $u(0, x^i)$, then the problem is well-posed. In other words, the norm of the solutions of this problem can not increase more rapidly than with exponential rate. Not all hyperbolic systems satisfy this criterion.

Definition 3.1. Depending on the criteria for the *principal part* or *principal symbol* or *characteristic matrix* $A(u)$ of the equation we distinguish the following types of hyperbolicity:

- *symmetric hyperbolic:* $A(u)$ can be symmetrized.
- *strongly hyperbolic:* $A(u)$ has real eigenvalues and a complete set of eigenvectors.
- *weakly hyperbolic:* $A(u)$ has real eigenvalues but not a complete set of eigenvectors.

Symmetric hyperbolicity implies strong hyperbolicity. The prototypical example for a symmetric hyperbolic system is again the wave equation.

The influence of the principal symbol on the well-posedness is discussed in [22, chapter 2 and 3]. The important result is, that strongly hyperbolic systems, and therefore the symmetric hyperbolic ones too, are well-posed, whereas the weakly hyperbolic ones are not.

It turns out, that the ADM formulation in Equation 2.64 is only weakly hyperbolic. By adding constraint terms to this formulation and the transition to the GBSSN formulation we get a strongly hyperbolic system. This property is preserved under the spherical reduction, so the system in Equation 2.83 with the lapse in Equation 2.89 and the shift in Equation 2.92 is strongly hyperbolic.

This is the place for some words of caution: The proof of well-posedness for strongly hyperbolic systems used the principal part only. Nevertheless, the source term $S(u)$ may also lead to contributions to the solution which increase faster than exponential. For numerical simulations the well-posedness is necessary but not sufficient. Even exponential growth may turn out to be too much for any numerical simulation to give meaningful results.

Following an analogous discussion in [15, p. 574], we denote the eigenvalues of the $m \times m$-matrix $A(u)$ by

$$\lambda_1(u),\ \lambda_2(u),\ \ldots,\ \lambda_m(u). \tag{3.3}$$

Then we get:

Definition 3.2. For each $k = 1, \ldots, m$ let $r_k(u)$ be the nonzero *right eigenvector* of $A(u)$ corresponding to $\lambda_k(u)$ and satisfying the equation

$$A(u) \cdot r_k(u) = \lambda_k(u) \cdot r_k(u). \tag{3.4}$$

Strong hyperbolicity implies that $\{r_k(u)\}_{k=1}^m$ is complete. Analogously we get:

Definition 3.3. For each $k = 1, \ldots, m$ let $l_k(u)$ be the nonzero *left eigenvector* of $A(u)$ corresponding to $\lambda_k(u)$ and satisfying the equation

$$l_k(u) \cdot A(u) = \lambda_k(u) \cdot l_k(u). \tag{3.5}$$

Equivalently this condition may be written as

$$A(u)^T \cdot l_k(u)^T = \lambda_k(u) \cdot l_k(u)^T. \tag{3.6}$$

Again the set $\{l_k(u)\}_{k=1}^m$ is complete.

We define the matrix V by taking the eigenvectors $\{r_k(u)\}_{k=1}^m$ or $\{l_k(u)^T\}_{k=1}^m$ as column vectors. So we can diagonalize A or A^T by

$$A = V \Lambda V^{-1} \qquad \text{or} \qquad A^T = V \Lambda V^{-1} \tag{3.7}$$

where Λ denotes the diagonal matrix with the eigenvalues.

Definition 3.4. Consider the situation above. We call

$$(X_1, X_2, \ldots, X_m) = X = V^{-1}u \tag{3.8}$$

the *characteristic (eigen)fields* X_k of the PDE system given by Equation 3.1. The associated eigenvalues are called *characteristic speeds*.

The choice of the eigenvector sets used in the diagonalization is a matter of practical convenience. Our choices will depend on the simplest form of resulting characteristic fields. Even more, one should remember that the eigenvectors are only unique up to a scalar factor. So the formulas for the characteristic fields may be simplified by using this freedom of choice.

3.2 The one dimensional wave equation

A simple example of a symmetric (and therefore strongly) hyperbolic system is the one dimensional wave equation, which can be defined as a second order partial differential equation

$$\Box\phi = -\frac{\partial^2\phi}{\partial t^2} + \frac{\partial^2\phi}{\partial x^2} = 0. \tag{3.9}$$

By the introduction of new field variables which are defined as the first time and spatial derivatives of ϕ via

$$\pi := \partial_t\phi \qquad \psi := \partial_x\phi \qquad u := (\pi, \psi)^T \tag{3.10}$$

we arrive at the first order system:

$$\partial_t u + A \cdot \partial_x u = 0 \qquad \text{with} \qquad A = -\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{3.11}$$

The eigenvalues of the matrix A are

$$\lambda_1 = 1 \quad \text{and} \quad \lambda_2 = -1 \tag{3.12}$$

with corresponding right eigenvectors

$$r_1 = (1, -1) \quad \text{and} \quad r_2 = (1, 1). \tag{3.13}$$

So the matrix V for the diagonalization and its inverse are given by

$$V = \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix} \qquad V^{-1} = \frac{1}{2} \cdot \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix} \tag{3.14}$$

which leads to:

Field	Expression
X_1	$\frac{1}{2} \cdot (\pi - \psi)$
X_2	$\frac{1}{2} \cdot (\pi + \psi)$

Table 3.1: Characteristic fields of the one dimensional wave equation.

For later calculations we require also expressions for the fundamental fields in terms of the characteristic fields. They are given by:

Field	Expression
π	$X_2 + X_1$
ψ	$X_2 - X_1$

Table 3.2: Fundamental fields in terms of characteristic fields for the wave equation.

3.3 The wave equation in spherical symmetry

The second toy model we consider is the three-dimensional wave equation in spherical symmetry. The aim is to test the influences on a hybrid grid caused by inner boundary conditions at the origin and increasing amplitudes and gradients for small values of the radial coordinate.

Following [15, p. 70] this equation is also known as the Euler-Poisson-Darboux equation for dimension $n = 3$ and given by:

$$\frac{\partial^2 \phi}{\partial t^2} - \frac{\partial^2 \phi}{\partial r^2} - \frac{2}{r} \cdot \frac{\partial \phi}{\partial r} = 0 \qquad \text{in} \quad \mathbb{R}_+ \times (0, \infty) \tag{3.15}$$

It is called wave equation because in spherical coordinates the Laplacian operator on a pure radial function takes the form:

$$\triangle \phi = \frac{\partial^2 \phi}{\partial r^2} + \frac{2}{r} \cdot \frac{\partial \phi}{\partial r} \tag{3.16}$$

Again we introduce the fields

$$\pi := \partial_t \phi \qquad \psi := \partial_r \phi \qquad u := (\pi, \psi)^T \tag{3.17}$$

and rewrite the equation as a first order system:

$$\partial_t u + A \cdot \partial_x u = S(u) \qquad \text{with} \quad A = -\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \text{and} \quad S(u) = \begin{pmatrix} \frac{2}{r} \cdot \psi \\ 0 \end{pmatrix} \tag{3.18}$$

We see that we got the same matrix A as in the case of the one dimensional wave equation and conclude that the characteristic fields are the same too:

Field	Expression
X_1	$\frac{1}{2} \cdot (\pi - \psi)$
X_2	$\frac{1}{2} \cdot (\pi + \psi)$

Table 3.3: Characteristic fields of the wave equation in spherical symmetry.

3.4 Analysis of the GBSSN system in spherical symmetry

In the following we discuss the GBSSN system and extract its characteristic fields. We get the full system by combining the spherical GBSSN equations in Equation 2.83 with the lapse in Equation 2.89 and the shift in Equation 2.92. Following [16, p. 4] one can reformulate this second order system as a first order system by the introduction of new field variables:

$$Q_\chi := \chi' \qquad Q_{g_{rr}} := g'_{rr} \qquad Q_{g_{\theta\theta}} := g'_{\theta\theta} \qquad Q_\alpha := \alpha' \qquad Q_{\beta^r} := (\beta^r)' \tag{3.19}$$

This gives the field variable vector

$$u = (\chi, g_{rr}, g_{\theta\theta}, \alpha, \beta^r, B_r, A_{rr}, K, \Lambda^r, Q_\chi, Q_{g_{rr}}, Q_{g_{\theta\theta}}, Q_\alpha, Q_{\beta^r})^T , \tag{3.20}$$

allowing us to rewrite the first order system analogous to the wave equation in the following form:

$$\partial_t u + A \cdot \partial_r u = S(u) \tag{3.21}$$

Again the vector $S(u)$ denotes terms without derivatives. The matrix A can be decomposed into the following blocks

$$A = \begin{pmatrix} 0_{5\times5} & 0_{5\times9} \\ 0_{9\times5} & \tilde{A} \end{pmatrix} \tag{3.22}$$

where $\tilde{A}$ takes the form:

$$\begin{pmatrix}
-\beta^r & 0 & \frac{\alpha}{g_{rr}} & 0 & 0 & -\frac{\beta^r v}{8g_{rr}^2} & -\frac{\beta^r v}{4g_{rr}g_{\theta\theta}} & 0 & -\frac{v+3}{4g_{rr}} \\
0 & -\beta^r & 0 & -\frac{2}{3}\alpha\chi g_{rr} & -\frac{\alpha}{3} & \frac{\alpha\chi}{3g_{rr}} & -\frac{\alpha\chi}{3g_{\theta\theta}} & \frac{2\chi}{3} & 0 \\
0 & 0 & -\beta^r & 0 & 0 & 0 & 0 & \frac{\chi}{g_{rr}} & 0 \\
0 & 0 & \frac{4\alpha}{3g_{rr}} & -\beta^r & 0 & -\frac{\beta^r v}{6g_{rr}^2} & -\frac{\beta^r v}{3g_{rr}g_{\theta\theta}} & 0 & -\frac{v+3}{3g_{rr}} \\
0 & 0 & -\frac{2\alpha\chi}{3} & 0 & -\beta^r & \frac{\beta^r\chi v}{3g_{rr}} & \frac{2\beta^r\chi v}{3g_{\theta\theta}} & 0 & \frac{2\chi v}{3} \\
0 & 2\alpha & 0 & 0 & 0 & \frac{\beta^r v}{3} - \beta^r & \frac{2\beta^r g_{rr} v}{3g_{\theta\theta}} & 0 & \frac{2g_{rr}v}{3} - 2g_{rr} \\
0 & -\frac{\alpha g_{\theta\theta}}{g_{rr}} & 0 & 0 & 0 & \frac{\beta^r g_{\theta\theta} v}{3g_{rr}} & \frac{1}{3}(2\beta^r v - 3\beta^r) & 0 & \frac{2g_{\theta\theta}v}{3} \\
0 & 0 & \operatorname{ch}\alpha^2 + 2\operatorname{cu}\alpha & 0 & 0 & 0 & 0 & -\beta^r & 0 \\
-\frac{3}{4} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\beta^r
\end{pmatrix} \tag{3.23}$$

As in the case of the wave equation in spherical symmetry, we restrict our considerations to the principal part of the first order system. The upper left block of the matrix A leads to trivial characteristic fields only and can therefore be ignored. We consider the subsystem

$$\partial_t u_{6:14} + \tilde{A}\left(u_{1:5}\right) \cdot \partial_r u_{6:14} = 0 \tag{3.24}$$

only, where the index $i:j$ refers to the components i to j of the vector u. Now we calculate the eigenvalues of the matrix $\tilde{A}$:

Eigenvalue	**Expression**
λ_1	0
λ_2	$-\beta^r$
λ_3	$-\beta^r$
λ_4	$-\beta^r - \frac{\sqrt{3}}{2\sqrt{g_{rr}}}$
λ_5	$\frac{\sqrt{3}}{2\sqrt{g_{rr}}} - \beta^r$
λ_6	$-\frac{\sqrt{2}\sqrt{\alpha}\sqrt{\chi}}{\sqrt{g_{rr}}} - \beta^r$
λ_7	$\frac{\sqrt{2}\sqrt{\alpha}\sqrt{\chi}}{\sqrt{g_{rr}}} - \beta^r$
λ_8	$-\frac{\alpha\sqrt{\chi}}{\sqrt{g_{rr}}} - \beta^r$
λ_9	$\frac{\alpha\sqrt{\chi}}{\sqrt{g_{rr}}} - \beta^r$

Table 3.4: Eigenvalues of $\tilde{A}$.

The corresponding left eigenvectors are shown in the following table. Here we multiplied the eigenvectors by a scalar factor to get simpler expressions. This is also the reason for the choice of the left eigenvectors.

L.e.	**Expression**
l_1	$\{0,0,0,0,0,g_{\theta\theta},2g_{rr},0,0\}$
l_2	$\left\{\frac{4g_{rr}}{3},0,0,0,\frac{2}{\chi},-\frac{1}{2g_{rr}},-\frac{1}{g_{\theta\theta}},0,0\right\}$
l_3	$\left\{-\frac{4}{3},0,0,1,0,0,0,0,0\right\}$
l_4	$\left\{-g_{rr},0,\frac{\sqrt{3}\alpha\sqrt{g_{rr}}}{2\left(\frac{3}{4}-2\alpha\chi\right)},0,0,-\frac{2\sqrt{g_{rr}}}{\sqrt{3}\left(\frac{8g_{rr}}{\beta^r}+\frac{16g_{rr}^{3/2}}{\sqrt{3}}\right)},-\frac{2\sqrt{g_{rr}}}{\sqrt{3}\left(\frac{4g_{\theta\theta}}{\beta^r}+\frac{8\sqrt{g_{rr}}g_{\theta\theta}}{\sqrt{3}}\right)},-\frac{\alpha\chi}{\frac{3}{4}-2\alpha\chi},-\frac{2\sqrt{g_{rr}}}{\sqrt{3}}\right\}$
l_5	$\left\{-g_{rr},0,-\frac{\sqrt{3}\alpha\sqrt{g_{rr}}}{2\left(\frac{3}{4}-2\alpha\chi\right)},0,0,\frac{2\sqrt{g_{rr}}}{\sqrt{3}\left(\frac{8g_{rr}}{\beta^r}-\frac{16g_{rr}^{3/2}}{\sqrt{3}}\right)},\frac{2\sqrt{g_{rr}}}{\sqrt{3}\left(\frac{4g_{\theta\theta}}{\beta^r}-\frac{8\sqrt{g_{rr}}g_{\theta\theta}}{\sqrt{3}}\right)},-\frac{\alpha\chi}{\frac{3}{4}-2\alpha\chi},\frac{2\sqrt{g_{rr}}}{\sqrt{3}}\right\}$

l_6	$\left\{0, 0, -\frac{\sqrt{2}\sqrt{\alpha}\sqrt{g_{rr}}}{\sqrt{\chi}}, 0, 0, 0, 0, 1, 0\right\}$
l_7	$\left\{0, 0, \frac{\sqrt{2}\sqrt{\alpha}\sqrt{g_{rr}}}{\sqrt{\chi}}, 0, 0, 0, 0, 1, 0\right\}$
l_8	$\left\{0, \frac{3}{\sqrt{\chi}\sqrt{g_{rr}}}, -\frac{2\sqrt{g_{rr}}}{\sqrt{\chi}}, 2g_{rr}, \frac{1}{\chi}, -\frac{1}{g_{rr}}, \frac{1}{g_{\theta\theta}}, 0, 0\right\}$
l_9	$\left\{0, -\frac{3}{\sqrt{\chi}\sqrt{g_{rr}}}, \frac{2\sqrt{g_{rr}}}{\sqrt{\chi}}, 2g_{rr}, \frac{1}{\chi}, -\frac{1}{g_{rr}}, \frac{1}{g_{\theta\theta}}, 0, 0\right\}$

Table 3.5: Left eigenvectors of $\tilde{A}$.

As in the case of the wave equation, this leads to expressions for the characteristic fields, which we will use in the simulations for boundary and interface conditions.

Field	**Expression**
X_1	$2g_{rr}Q_{g_{\theta\theta}} + g_{\theta\theta}Q_{g_{rr}}$
X_2	$\frac{4B_r g_{rr}}{3} + \frac{2Q_\chi}{\chi} - \frac{Q_{g_{rr}}}{2g_{rr}} - \frac{Q_{g_{\theta\theta}}}{g_{\theta\theta}}$
X_3	$\Lambda^r - \frac{4B_r}{3}$
X_4	$\sqrt{g_{rr}}\left(\frac{2\left(\frac{3\alpha K}{4\left(\frac{3}{4}-2\alpha\chi\right)}-Q_{\beta^r}\right)}{\sqrt{3}} - \frac{\beta^r Q_{g_{\theta\theta}}}{4\beta^r\sqrt{g_{rr}}g_{\theta\theta}+2\sqrt{3}g_{\theta\theta}}\right) - \frac{\alpha\chi Q_\alpha}{\frac{3}{4}-2\alpha\chi} - \frac{\beta^r Q_{g_{rr}}}{8\left(\beta^r g_{rr}+\frac{\sqrt{3}\sqrt{g_{rr}}}{2}\right)} - B_r g_{rr}$
X_5	$\sqrt{g_{rr}}\left(\frac{2\left(Q_{\beta^r}-\frac{3\alpha K}{4\left(\frac{3}{4}-2\alpha\chi\right)}\right)}{\sqrt{3}} + \frac{\beta^r Q_{g_{\theta\theta}}}{2\sqrt{3}g_{\theta\theta}-4\beta^r\sqrt{g_{rr}}g_{\theta\theta}}\right) - \frac{\alpha\chi Q_\alpha}{\frac{3}{4}-2\alpha\chi} + \frac{\beta^r Q_{g_{rr}}}{4\sqrt{3}\sqrt{g_{rr}}-8\beta^r g_{rr}} - B_r g_{rr}$
X_6	$Q_\alpha - \frac{\sqrt{2}\sqrt{\alpha}\sqrt{g_{rr}}K}{\sqrt{\chi}}$
X_7	$\frac{\sqrt{2}\sqrt{\alpha}\sqrt{g_{rr}}K}{\sqrt{\chi}} + Q_\alpha$
X_8	$\frac{\frac{\sqrt{g_{rr}}(3A_{rr}-2g_{rr}K)}{\sqrt{\chi}}+2g_{rr}^2\Lambda^r-Q_{g_{rr}}}{g_{rr}} + \frac{Q_\chi}{\chi} + \frac{Q_{g_{\theta\theta}}}{g_{\theta\theta}}$
X_9	$\frac{\frac{\sqrt{g_{rr}}(2g_{rr}K-3A_{rr})}{\sqrt{\chi}}+2g_{rr}^2\Lambda^r-Q_{g_{rr}}}{g_{rr}} + \frac{Q_\chi}{\chi} + \frac{Q_{g_{\theta\theta}}}{g_{\theta\theta}}$

Table 3.6: Characteristic fields of the GBSSN equation.

We will also require formulas for the fundamental fields depending on the characteristic ones.

Field	Expression
B_r	$\frac{4g_{\theta\theta}\left(\frac{3}{4}-(\beta^r)^2 g_{rr}\right)\left(\alpha\chi(-2X_4-2X_5+X_6+X_7)+\frac{3(X_4+X_5)}{4}\right)-(\beta^r)^2 X_1\left(\frac{3}{4}-2\alpha\chi\right)}{8g_{rr}g_{\theta\theta}\left(\frac{3}{4}-2\alpha\chi\right)\left((\beta^r)^2 g_{rr}-\frac{3}{4}\right)}$
A_{rr}	$\frac{\sqrt{\chi}\sqrt{g_{rr}}\left(\sqrt{2}\sqrt{\alpha}(X_8-X_9)-2X_6+2X_7\right)}{6\sqrt{2}\sqrt{\alpha}}$
K	$\frac{\sqrt{\chi}(X_7-X_6)}{2\sqrt{2}\sqrt{\alpha}\sqrt{g_{rr}}}$
Λ^r	$\frac{2\alpha\chi\left(2g_{\theta\theta}\left(\frac{3}{4}-(\beta^r)^2 g_{rr}\right)(3g_{rr}X_3-2X_4-2X_5+X_6+X_7)+(\beta^r)^2 X_1\right)+\frac{3}{4}\left(2g_{\theta\theta}\left((\beta^r)^2 g_{rr}-\frac{3}{4}\right)(3g_{rr}X_3-2(X_4+X_5))-(\beta^r)^2 X_1\right)}{6g_{rr}g_{\theta\theta}\left(\frac{3}{4}-2\alpha\chi\right)\left((\beta^r)^2 g_{rr}-\frac{3}{4}\right)}$
Q_χ	$\frac{\chi\left(2g_{rr}g_{\theta\theta}\left((\beta^r)^2 g_{rr}-\frac{3}{4}\right)\left(2\alpha\chi(-3X_2-2X_4-2X_5+X_6+X_7)+\frac{3}{4}(3X_2+2(X_4+X_5))\right)-X_1\left(\frac{3}{4}-2\alpha\chi\right)\left(\frac{9}{4}-4(\beta^r)^2 g_{rr}\right)\right)}{12g_{rr}g_{\theta\theta}\left(\frac{3}{4}-2\alpha\chi\right)\left((\beta^r)^2 g_{rr}-\frac{3}{4}\right)}$
$Q_{g_{rr}}$	$\frac{2g_{rr}g_{\theta\theta}\left(\frac{3}{4}-(\beta^r)^2 g_{rr}\right)\left(-2\alpha\chi(4g_{rr}X_3+X_2-2X_4-2X_5+X_6+X_7-X_8-X_9)-\frac{3}{4}(-4g_{rr}X_3-X_2+2X_4+2X_5+X_8+X_9)\right)+X_1\left(\frac{3}{4}-2\alpha\chi\right)\left(\frac{9}{4}-2(\beta^r)^2 g_{rr}\right)}{6g_{\theta\theta}\left(\frac{3}{4}-2\alpha\chi\right)\left(\frac{3}{4}-(\beta^r)^2 g_{rr}\right)}$
$Q_{g_{\theta\theta}}$	$\frac{2g_{rr}g_{\theta\theta}\left(\frac{3}{4}-(\beta^r)^2 g_{rr}\right)\left(-2\alpha\chi(4g_{rr}X_3+X_2-2X_4-2X_5+X_6+X_7-X_8-X_9)-\frac{3}{4}(-4g_{rr}X_3-X_2+2X_4+2X_5+X_8+X_9)\right)-X_1\left(\frac{3}{4}-2\alpha\chi\right)\left(\frac{9}{4}-4(\beta^r)^2 g_{rr}\right)}{12g_{rr}\left(\frac{3}{4}-2\alpha\chi\right)\left((\beta^r)^2 g_{rr}-\frac{3}{4}\right)}$
Q_α	$\frac{X_6+X_7}{2}$
Q_{β^r}	$\frac{\frac{3\sqrt{\alpha}\sqrt{\chi}\sqrt{g_{rr}}(X_7-X_6)}{\sqrt{2}\left(\frac{3}{4}-2\alpha\chi\right)}+\frac{2\sqrt{3}(\beta^r)^2 g_{rr}^{3/2}g_{\theta\theta}(X_5-X_4)+\frac{3\beta^r X_1}{4}+\frac{3}{2}\sqrt{3}\sqrt{g_{rr}}g_{\theta\theta}(X_4-X_5)}{g_{\theta\theta}\left((\beta^r)^2 g_{rr}-\frac{3}{4}\right)}}{8g_{rr}}$

Table 3.7: Fundamental fields depending on characteristic fields of the GBSSN equation.

4 Simulations of the one dimensional wave equation

As described in the previous chapter, we deal with hyperbolic systems of partial differential equations. The typical example is the one dimensional wave equation. Our aim is to construct hybrid numerical methods for the simulation of the GBSSN system. To achieve this, we develop and test these methods for the case of the simple wave equation first. Most of the so created code fragments can then be reused for the GBSSN implementation.

4.1 Implemented boundary conditions

4.1.1 Radiation boundary conditions

The radiation boundary conditions are implemented using the characteristic fields given in Table 3.1. First we consider the left boundary. We calculate the characteristic variable for the outgoing characteristic field (in this case to the eigenvalue $\lambda_2 = -1$ by Equation 3.12) and denote array indices for discrete evolution variables by square brackets [.]:

$$X_2[0] = \frac{1}{2}\left(\pi[0] + \psi[0]\right) \tag{4.1}$$

Here the array index 0 denotes the first element of the grid on the left boundary.

Then we assume that the ingoing mode $X_1[0]$, associated to the eigenvalue $\lambda_1 = 1$, has to be zero. This realizes the condition that all wave components reaching the boundary should leave the interval without any reflections.

By Table 3.2, in this case the value of $X_2[0]$ coincides with the values we have to set for the fundamental fields:

$$\pi[0] = X_2[0] \qquad \psi[0] = X_2[0] \tag{4.2}$$

Interchanging ingoing and outgoing mode at the right boundary leads to

$$\pi[-1] = X_1[-1] \qquad \psi[-1] = -X_1[-1] \tag{4.3}$$

where the array index -1 denotes the last element of the grid on the right boundary.

4.1.2 Interface conditions

Inspired by the idea of the implementation of radiation boundary conditions introduced above, we want to derive conditions for the interface in the hybrid grid again by using characteristic fields of the wave equation. The main idea of this approach is to use the function values of the left grid segment at the interface point to calculate the right going characteristic field ($\lambda_1 = 1$). Analogously we use the function values of the right grid segment to calculate the left going characteristic field ($\lambda_2 = -1$). This can be formulated as

$$X_1[\text{if}] = \frac{1}{2}\left(\pi[\text{l}] + \psi[\text{l}]\right) \tag{4.4a}$$

$$X_2[\text{if}] = \frac{1}{2}\left(\pi[\text{r}] + \psi[\text{r}]\right) \tag{4.4b}$$

where the indices 'l' and 'r' denote the values from the left and right grid segment and 'if' denotes the common interface point. By using Table 3.2 we get as new values for the fundamental fields:

$$\pi[\text{l}] = \pi[\text{r}] = X_2[\text{if}] + X_1[\text{if}] \tag{4.5a}$$

$$\psi[\text{l}] = \psi[\text{r}] = X_2[\text{if}] - X_1[\text{if}] \tag{4.5b}$$

This works fine for the first order system. Because the GBSSN system, which we will consider later, is of second order in space, we want to formulate these conditions with second order variables only. For the wave equation, we just have to drop the field ψ. Then the system reads:

$$\partial_t \phi = \pi \tag{4.6a}$$

$$\partial_t \pi = (\partial_x)^2 \phi \tag{4.6b}$$

But now we are missing the variable ψ for the calculation of the interface conditions. This problem can be solved by calculating the first spatial derivative of our physical field ϕ with the appropriate differentiation method (either FD or PS). Then we insert this result for the variable ψ in Equation 4.4.

The rest of the algorithm for the interface calculation remains the same, except setting the fundamental fields. In the first order system we were able to set a value for the variable ψ. This is not possible anymore. But we observe, that in Equation 4.6 the right hand side of the time derivative of ϕ is only the π variable. For this variable we already set the interface conditions previously, so there is nothing left to do.

For the GBSSN system we will have to find a method to incorporate the conditions for the spatial derivative of the remaining variables.

4.2 Test methods

In the following we will introduce methods to measure the numerical errors which occur during the simulation process.

4.2.1 Comparison with the analytic solution

This is only applicable to very simple equations and initial values where of course the analytic solution is known. The aim is to check the implementation of the simulation methods. In the case of the one dimensional wave equation we get by [15, p. 68] that the general solution of

$$\frac{\partial^2 \phi}{\partial t^2} - \frac{\partial^2 \phi}{\partial x^2} = 0 \qquad \text{in} \quad \mathbb{R} \times (0, \infty) \tag{4.7}$$

satisfying

$$\phi = g, \quad \partial_t \phi = h \qquad \text{on} \quad \mathbb{R} \times \{t = 0\} \tag{4.8}$$

is given by *d' Alembert's formula*:

$$\phi(x,t) = \frac{1}{2}\left(g(x+t) + g(x-t)\right) + \frac{1}{2}\int_{x-t}^{x+t} h(y)dy \quad (x \in \mathbb{R},\ t \geq 0) \tag{4.9}$$

So we can calculate the errors of our numerical simulation by comparing it to the analytic solution for arbitrary time steps from our initial data.

Because we have to deal with a finite interval, we have to take into account the boundary conditions. Instead of solving the wave equation for these special boundary conditions we can also choose simulation times which give final results where we know the analytic solution. Reflecting boundary conditions and a final time where the solution coincides with the initial data is an example of this approach. Another one is the implementation of radiation boundary conditions for a wave packet as initial data and the comparison of the final time slice with the zero function. Both were implemented during the test period.

If we calculate the errors of simulations with different resolutions we can check if the errors decrease with the expected rate.

4.2.2 Convergence factor

We have already introduced the main ideas of this concept in Section 1.6.3. Here we only want to mention that we can create plots which contain the differences between the simulation runs. Then we can add a plot which multiplies the result of the difference of the two finer runs with the expected convergence factor. In the ideal case we would see that this plot coincides with the difference plot of the two coarser runs.

4.2.3 Conservation laws

For the wave equation we can use energy conservation to check our simulation results. With the notation introduced in Section 3.2 we calculate the energy for one time slice via

$$E = \frac{1}{2} \cdot \int \pi^2 + \psi^2 \, dx \tag{4.10}$$

where we integrate over our physical interval.

To test our implementation we have to calculate the energy values for a significant set of time slices and verify if the values stay constant or drop of appropriately in the case of radiation boundary conditions.

4.3 Verification of the convergence orders in single domains

We start the evaluation of our discretization methods by showing that their implementation gives the expected order of convergence on a single interval. Therefore we choose as initial data a *Gaussian* given by:

$$\phi(x, t = 0) = A \cdot \exp\left[- \left(\frac{x - x_0}{\sigma} \right)^2 \right] \tag{4.11}$$

We call x_0 the *center* and σ the *width* of the function.

As a consequence, the initial data for the fields in the associated first order system is:

$$\pi(x, t = 0) = 0 \qquad \psi(x, t = 0) = -2 \cdot \left(\frac{x - x_0}{\sigma^2} \right) \cdot \phi(x, t = 0) \tag{4.12}$$

We choose a fixed time step size for all resolutions and radiation boundary conditions at both boundaries. As discussed in Section 4.1.1, the radiation boundary conditions use the function values at the outermost grid point. In case of the finite difference method this value is calculated by one-sided stencils of the same order as in the interior. For the PS method the same transformation from the spectral domain to the space domain as in the interior is used. Additional parameter choices are given in the following table:

dt	**phys. domain**	$\boldsymbol{A}$	$\boldsymbol{x_0}$	$\boldsymbol{\sigma}$
0.002	$[0, 2]$	0.05	1.5	0.1

Table 4.1: Simulation parameters for 4th order convergence evaluation.

First we show that our implementations of the second and fourth order finite difference methods work properly. We run the simulation until the two wave packets have left the physical domain at $t = 2$. Then we compare the resulting errors to the analytic solution, which is in this case the zero line. We do this by calculating the maximum absolute value of the function at the last time slice, i.e. we calculate the maximum norm (l_∞) at $t = 2$.

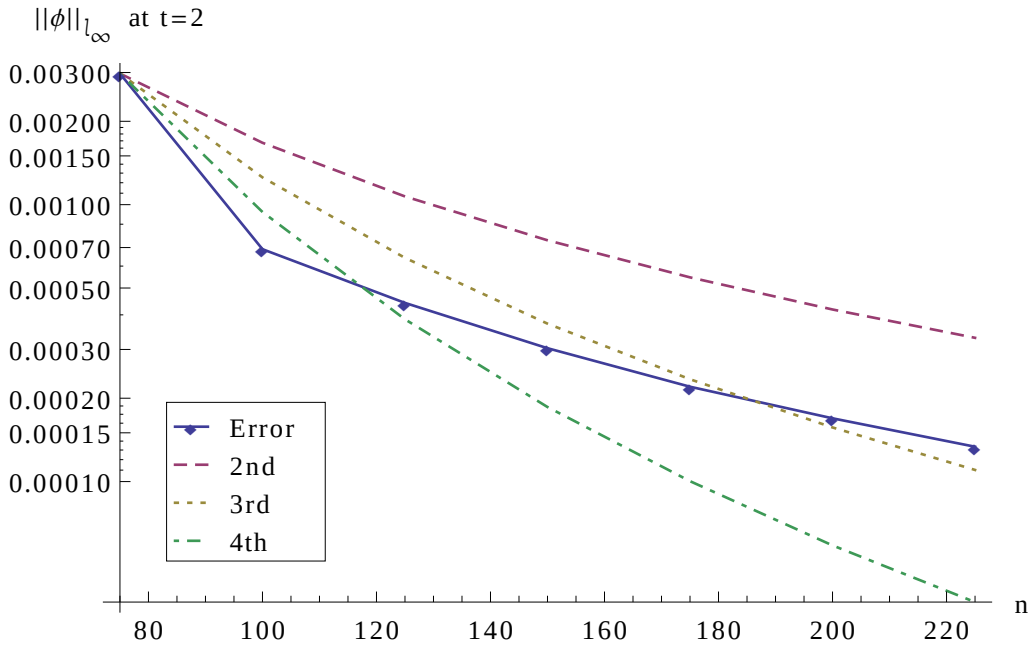


Figure 4.1: Second order finite difference method errors at $t = 2$. The thick line with plot markers indicates the simulation results. The other lines show second, third and fourth order convergence for comparison. As expected the errors decrease with better than second order.

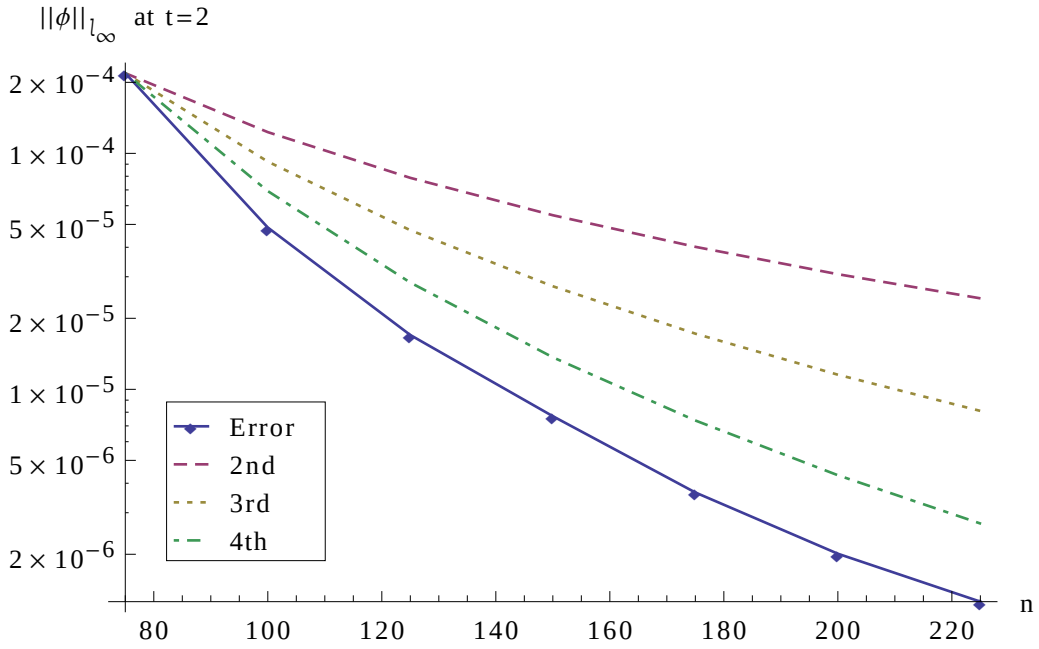


Figure 4.2: Fourth order finite difference method errors at $t = 2$. The thick line with plot markers indicates the simulation results. The other lines show second, third and fourth order convergence for comparison. The errors decrease with higher than fourth order.

Next we consider the results for a pseudospectral discretization using Chebyshev polynomials. First we run simulations with $dt = 0.001$ independent of the PS order. The time step is chosen such that the time-integration error is smaller than the spatial discretization error even for the highest number of gridpoints.

Next we run simulations with the same PS orders where we choose for each individual run a time step determined by the minimal grid spacing. So in general the varying time steps will be larger than $dt = 0.001$.

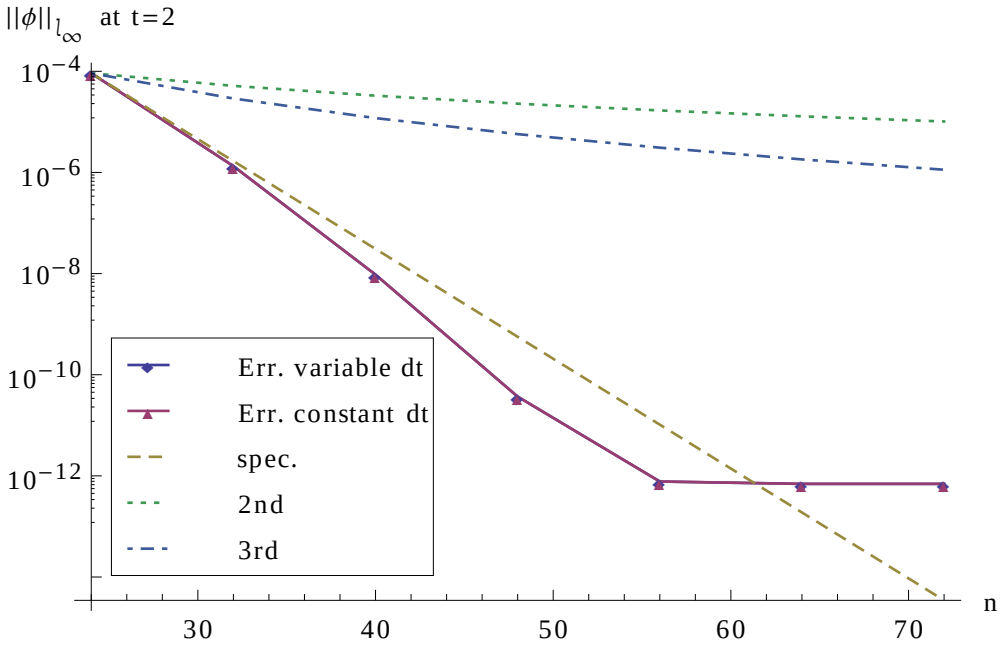


Figure 4.3: Pseudospectral method with Chebyshev polynomials. Error plot at $t = 2$ for variable and constant dt. For the comparison with a spectral convergence rate the function $\exp(n/2)$ was chosen.

We observe that the absolute values of the maximal errors do not change significantly. So in the case where the time step depends on the chosen pseudospectral order (this time step is bigger than the one in the simulations where it is equal for all orders), the errors introduced by the spatial approximation already dominate the errors introduced by the time stepping algorithm.

In the following, whenever we compare the simulation results of different FD resolutions and/or PS orders, we will use time steps dt equal in all runs to allow easier comparison of individual time slices.

Note that for some higher resolutions the errors do not further decrease. We refer the reader to Section 1.6.2 where we discussed the influence of machine sized numbers on the simulation.

4.4 Energy tests with the hybrid grid

The energy test with the hybrid grid compares the energy values for different resolutions calculated by Equation 4.10. The parameters used are listed in the following table. Half of the interval is implemented as a fourth order finite difference grid, the other half is a pseudospectral Chebyshev grid. We use radiation boundary conditions on both sides of the interval and the interface conditions introduced in Section 4.1.2 for the interface between the finite difference and pseudospectral grid segment.

dt	phys. domain	A	x_0	σ	FD pts.	PS orders
0.002	$[0, 2]$	0.05	1.5	0.1	162-365-650	20-30-40

Table 4.2: Simulation parameters for energy tests.

The choices for the FD points depend on the PS orders. We want to run our simulations with a finite difference grid-spacing appropriately the same size as the minimum PS grid spacing.

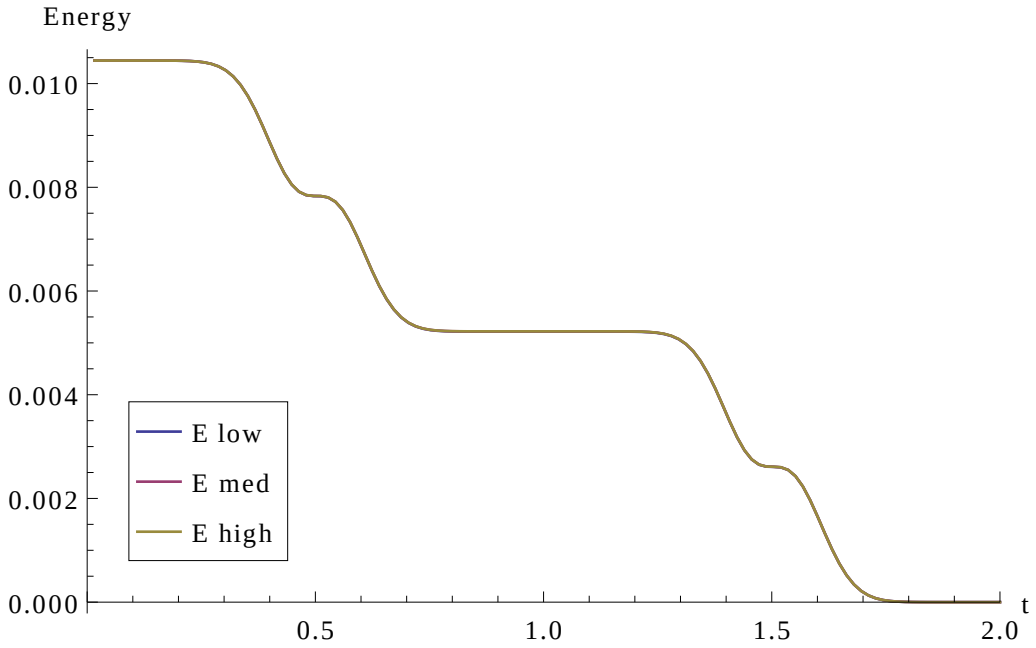


Figure 4.4: Energy values for different resolutions depending on time t.

The previous plot shows constant energy values for the time intervals where the wave packets travel through the interval. At the time values $t = 0.5$ and $t = 1.5$ the peak values of the packets leave the interval and the energy values drop as expected.

Next we consider the difference between the high and medium resolutions multiplied by the expected convergence factors for the fourth (cf4) order finite difference method and the

pseudospectral grids (cfPS), which are calculated by the formulas in Table 1.7. In this simulation example we calculate the following factors:

$$\text{cfPS} = 4.60 \qquad \text{cf4} = 27.50 \tag{4.13}$$

Note that in this situation the PS factor is much smaller than the fourth order finite difference factor. The formulas for the convergence factors depend only on the ratios between the chosen FD resolutions and PS orders respectively. Getting similar grid separations at the interface requires different resolution ratios for the FD and PS segments. In this example the ratios between the FD resolutions is approximately 2, but in the PS segment its only 1.5. Furthermore, these values fall into the interval where the polynomial x^4 dominates the exponential function.

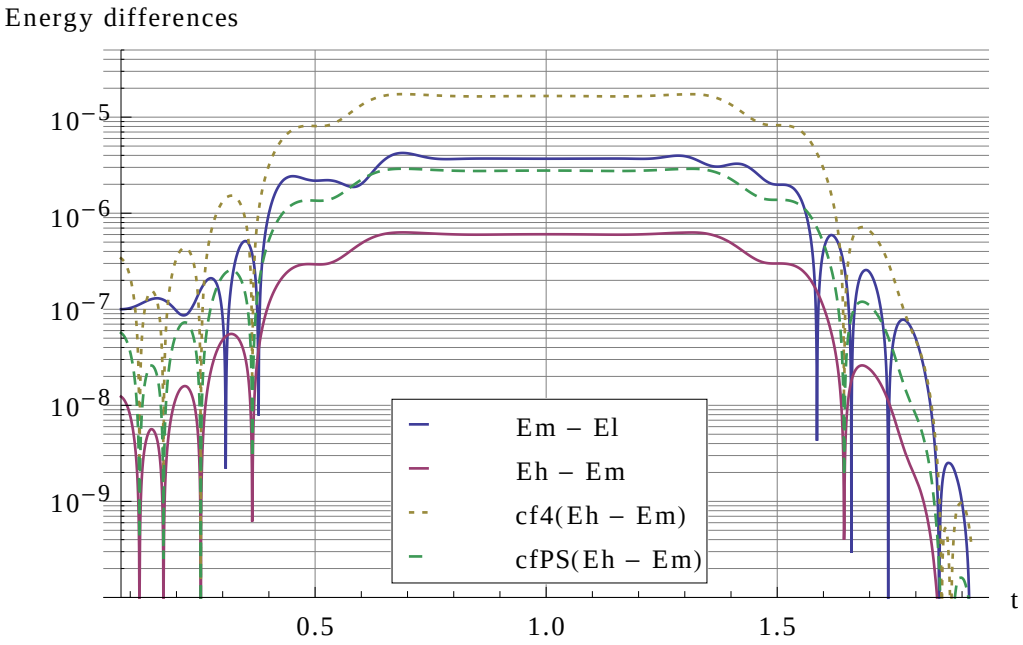


Figure 4.5: Energy differences depending on time t. The thick lines indicate the energy differences, the dashed lines the energy differences between the high and medium resolution multiplied by the expected convergence factors for the FD and PS segments.

As expected, the convergence order lies somewhere between the convergence order of the fourth order finite difference method and the pseudospectral Chebyshev method. In this case it is found to be close to the expected pseudospectral convergence rate.

4.5 Convergence order of the hybrid grid

We saw above that our results for the wave equation on the hybrid grid are reasonable. Now we want to evaluate whether the convergence order of the finite difference grid or the one of the pseudospectral grid dominates. We repeat the simulations of Section 4.3 with the hybrid grid, where we choose for the interval $[0, 1]$ a fourth order finite difference method and for the interval $[1, 2]$ a Chebyshev pseudospectral method.

The question arises, what value should be put on the horizontal axis to get reasonable results. We have no formula at our disposal to calculate an expected convergence order of a hybrid grid by taking the finite difference and pseudospectral resolutions into account. So the only two choices which make sense are the dependencies of the error at $t = 2$ on the finite difference number of points and the order of the pseudospectral grid. We expect the result for the hybrid grid to lie somewhere in between.

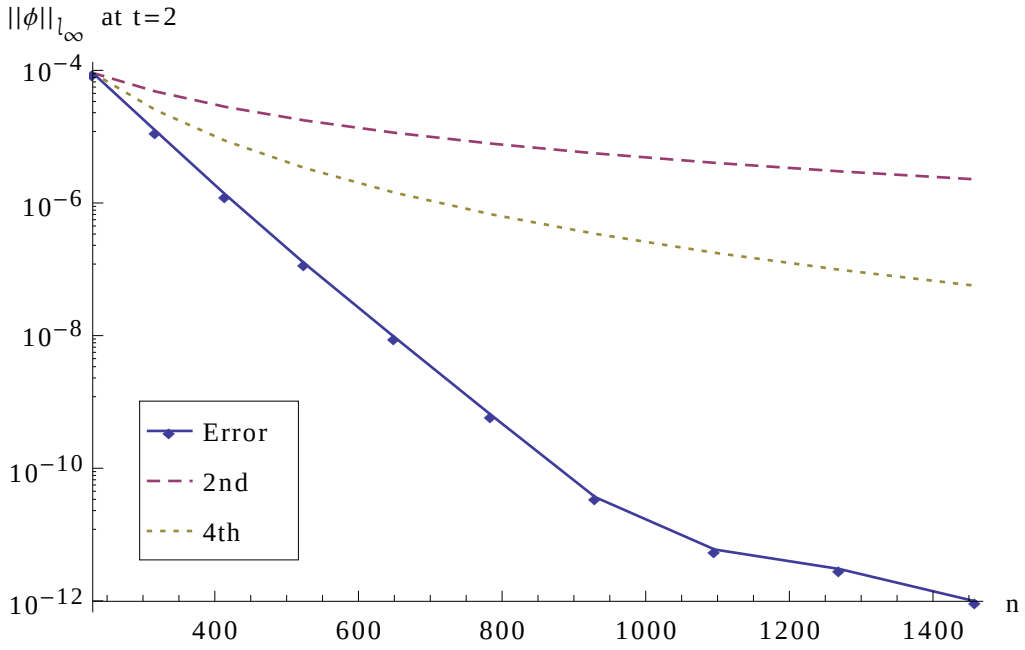


Figure 4.6: Error at $t = 2$ depending on the number of finite difference grid points compared to second and fourth order convergence.

The previous plot shows definitely better convergence than a usual fourth order finite difference scheme. Analogously, with the dependency on the pseudospectral order the plot is given in the following:

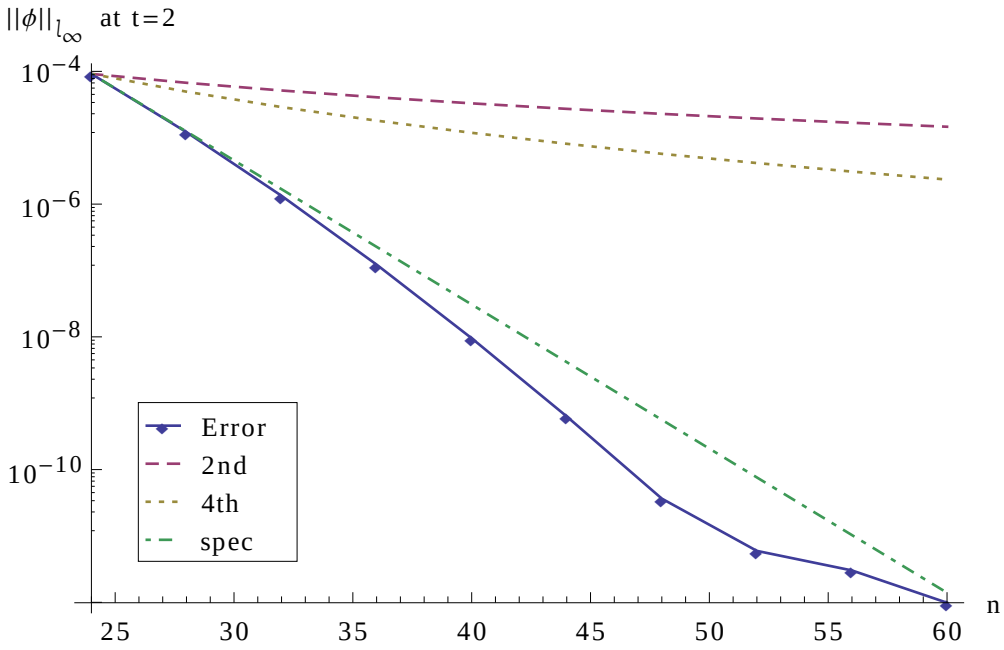


Figure 4.7: Error at $t = 2$ depending on the order of the pseudospectral grid segment compared to second, fourth and spectral order convergence. For the spectral convergence again the function $\exp(n/2)$ was chosen.

We observe that we did not loose much of the spectral convergence rate we had for the pseudospectral grid alone. Again the curve starts to flatten for higher orders when effects of machine sized values arise.

4.6 Conclusions for the one dimensional wave equation

- The position of the peak of the wave packet in the initial data was varied during the hybrid grid simulation test runs. The best results were achieved when the peak was initially located in the pseudospectral grid segment. That is not surprising if one considers the pseudospectral grid to have a higher order of accuracy. See Table 4.2 for the particular parameter choices in these simulations. When the initial peak was positioned in the finite difference grid segment a very small peak (about 10^{-9}) remained there for the rest of the simulation. This led to earlier cutoffs during the convergence order tests. In single finite difference domains this peak would have been dominated by the errors of the method. Choosing higher FD resolutions minimizes this problem.
- We also performed convergence tests for arbitrary time slices during the test runs by using the analytic solution for comparison. Usually they started with a spectral rate and then showed a much earlier cutoff with larger errors remaining for all higher orders. However, the results depended on the selected time slice and the position of the wave packets at those times (inside of both grid segments, interface, boundary).

5 Wave equation in spherical symmetry

After running tests with the one-dimensional wave equation in the previous chapter, we will now consider the wave equation in spherical symmetry as an additional toy model. Here we will deal for the first time with a problem in spherical symmetry. Accordingly, we will have to adapt our methods to problems at the point with radius zero and find a way to formulate the inner boundary conditions there.

5.1 Boundary and interface conditions

We start by considering the boundary conditions applicable to this problem.

5.1.1 Inner boundary conditions

One of the terms in

$$\frac{\partial^2 \phi}{\partial t^2} - \frac{\partial^2 \phi}{\partial r^2} - \frac{2}{r} \cdot \frac{\partial \phi}{\partial r} = 0 \tag{5.1}$$

has the radius variable r in the denominator. This causes problems at the origin $r = 0$. To avoid that, we can switch to a cell-centered grid (see Section 1.1.1) in the finite difference regime. Such terms also occur in the GBSSN system. We will also use a cell-centered grid there.

With that in mind we impose a symmetry condition on the variable π at the origin. This works by introducing *ghost points* with negative r values. Their number depends on the order of the finite difference method. In case of a fourth order finite difference grid we require two ghost points.

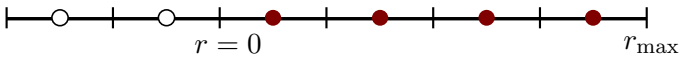


Figure 5.1: Illustration of ghost points at the left boundary of a cell-centered grid.

To impose symmetry at the origin, the values of the ghost points are set equal to their counterparts with positive r for the variable π. We do not set boundary conditions for the ϕ variable, as we calculate it by integrating π, for which boundary conditions are set already.

5.1.2 Other conditions

As in the simulation runs for the one-dimensional wave equation, we will use interface conditions and radiation boundary conditions. Due to the fact that the principal part of the wave equation in spherical symmetry remains unchanged compared to the one-dimensional equation, the same holds for the characteristic fields as mentioned in Section 3.3.

This consideration leads to the same conditions for the interface and the outer boundary as discussed in Section 4.1.2 and Section 4.1.1.

Nevertheless it is important to mention that with the choice of the cell centered finite difference grid one has to change the location of the first PS point. Otherwise the interface points would not be identified anymore and lead to larger errors.

5.2 Test methods

In Chapter 4 we compared the simulation results of the wave equation to an analytic solution. We repeat this process now for the wave equation in spherical symmetry. From [15, p. 71] we know that we get an analytic solution for the wave equation in spherical symmetry by dividing a solution for the one-dimensional wave equation by r.

So we conclude that the initial data

$$\phi(x, t = 0) = \frac{A}{r} \cdot e^{-\left(\frac{r-x_0}{\sigma}\right)^2} \tag{5.2}$$

leads to the analytic solution:

$$\phi(x, t) = \frac{A}{2r} \cdot \left[e^{-\left(\frac{r+t-x_0}{\sigma}\right)^2} + e^{-\left(\frac{r-t-x_0}{\sigma}\right)^2} \right] \tag{5.3}$$

Note that we did not incorporate the boundary conditions in this calculation. Hence the analytic solution is only valid for the iteration steps until the ingoing wave packet hits the inner boundary.

Additionally we have to modify the formula for the calculation of the energy, which now reads:

$$E = \frac{1}{2} \cdot \int \left(\pi^2 + \psi^2\right) \cdot r^2 \, dx \tag{5.4}$$

The method of comparing the results obtained with the expected convergence factor remains unchanged, as it is purely numerical and does not depend on the system.

5.3 Energy tests for the hybrid grid

As in the one-dimensional case we can calculate the energy for the simulation results of three runs with different resolutions.

The simulations run until $t = 7$ and the interface is located at $r = 0.5$. The results are plotted in the following diagram.

dt	phys. domain	A	x_0	σ	FD pts.	PS orders
0.0005	$[0, 4.5]$	0.05	1.5	0.25	125-180-325	50-60-80

Table 5.1: Simulation parameters for energy tests.

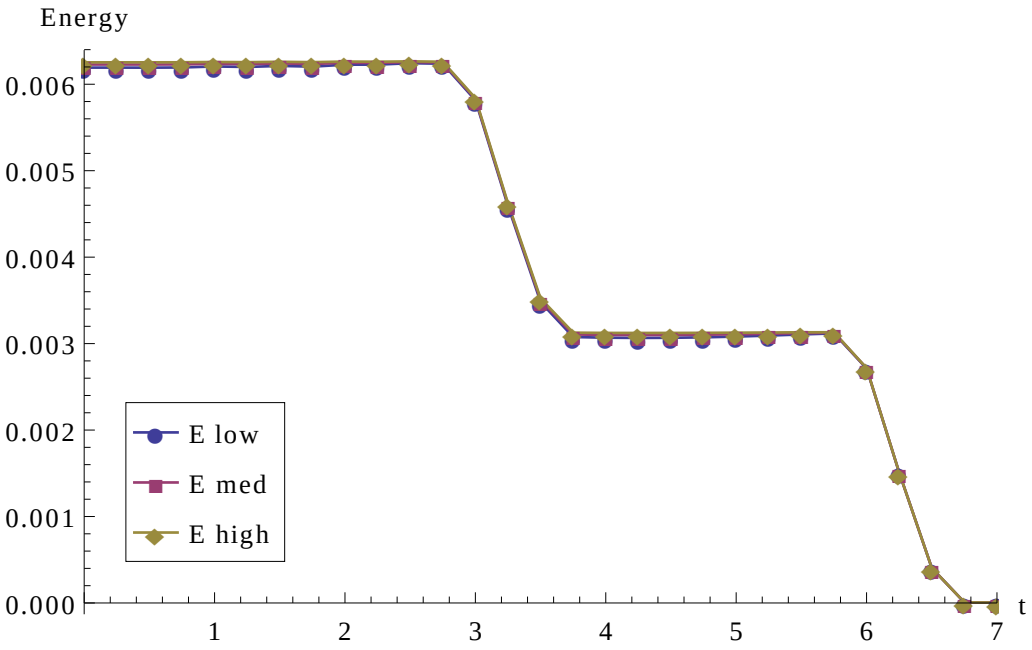


Figure 5.2: Energy values for different resolutions depending on time t.

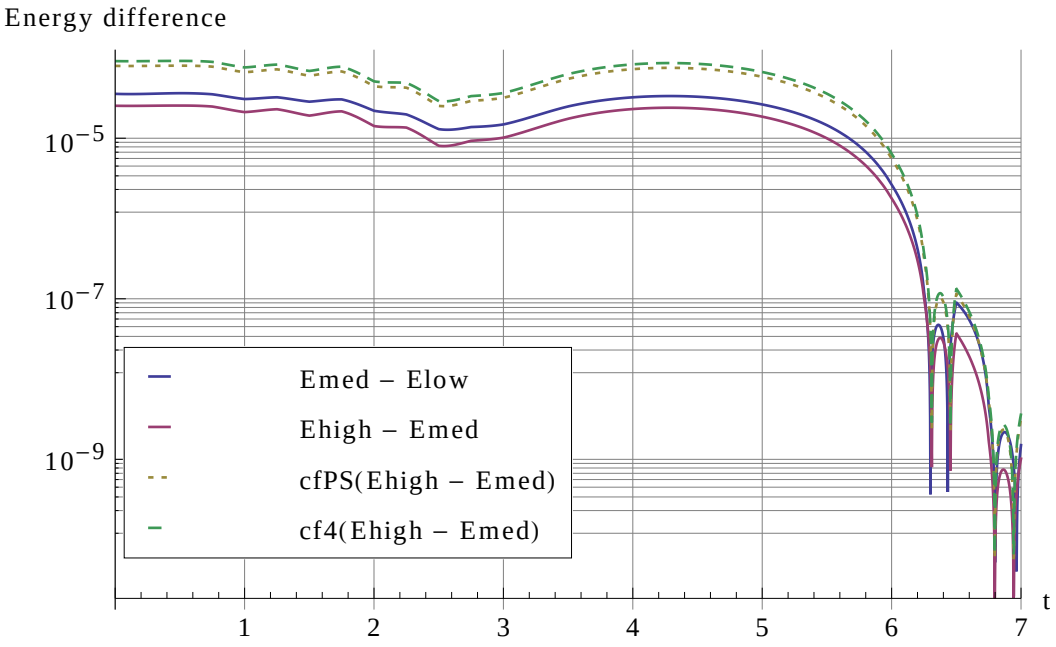


Figure 5.3: Energy differences depending on time t. Dashed and dotted lines show the values expected for fourth order finite difference and spectral methods.

Again we observe the loss of energy at the time values where the wave packets leave the interval. By considering the differences in the simulation runs we get convergence with a less than expected rate. We would estimate the convergence factor to lie between the one calculated for the pseudospectral and the fourth order finite difference grid.

5.4 Convergence test in variable ϕ

By using the simulation results from the previous energy test, we can also apply a convergence test to the variable ϕ for arbitrary time slices. We choose $t = 0.5$ and $t = 7$.

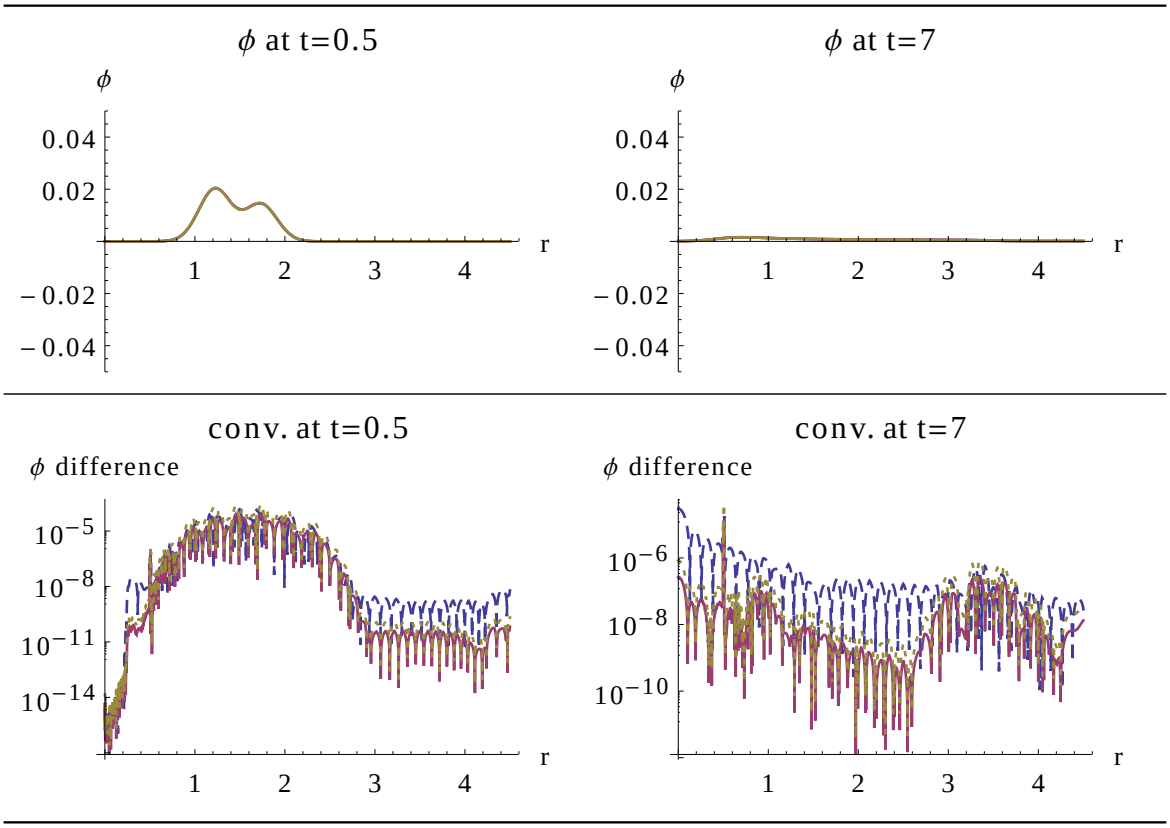


Table 5.2: The left column shows the function ϕ and its convergence test for $t = 0.5$ and the right column for $t = 7$. In each case the dashed line in the convergence plot indicates the difference between the lower resolutions, the thick one the difference between the higher resolutions and the dotted one the estimated values for spectral convergence.

In some areas where the function values are close to zero we see a convergence much better than the expected one. In the regions where the Gaussians are concentrated we do not reach spectral convergence but get approximately second order convergence calculated from the FD resolutions.

The function values of ϕ at $t = 7$ show how the errors of the outer boundary, caused by the outgoing wave packet, are amplified as they propagate towards $r = 0$.

5.5 Dependency on the finite difference order

In this section we want to verify how the choice of the finite difference grid spacing influences the simulation results. Therefore we will fix a pseudospectral grid order and vary the FD resolution in both directions.

5.5.1 Pseudospectral order 40

We start by stating our simulation parameters:

dt	**dom.**	$\boldsymbol{A}$	$\boldsymbol{x_0}$	$\boldsymbol{\sigma}$	**FD pts.**
0.0005	$[0, 4.5]$	0.05	1.5	0.25	$n \in \{10, 15, 20, 30, 40, 60, 80, 120, 160, 200, 240, 260\}$

Table 5.3: Simulation parameters for the dependency on the finite difference grid spacing for PS order 40.

The final time is fixed to $t = 7$ and the interface is still located at $r = 0.5$. The relation of the FD grid spacing and the minimum PS grid spacing will vary from 8 to approximately 0.3. We will check the errors at a time slice where we still have an analytic solution and at the final one, where both packets have left the interval. For the early time slice we choose $t = 0.75$, where the ingoing packet starts hitting the inner boundary and the analytic solution looses validity. Errors near to the boundary will be ignored throughout this discussion. In case of the simulation run with 80 FD points the minimum PS grid spacing approximately coincides with the FD grid spacing.

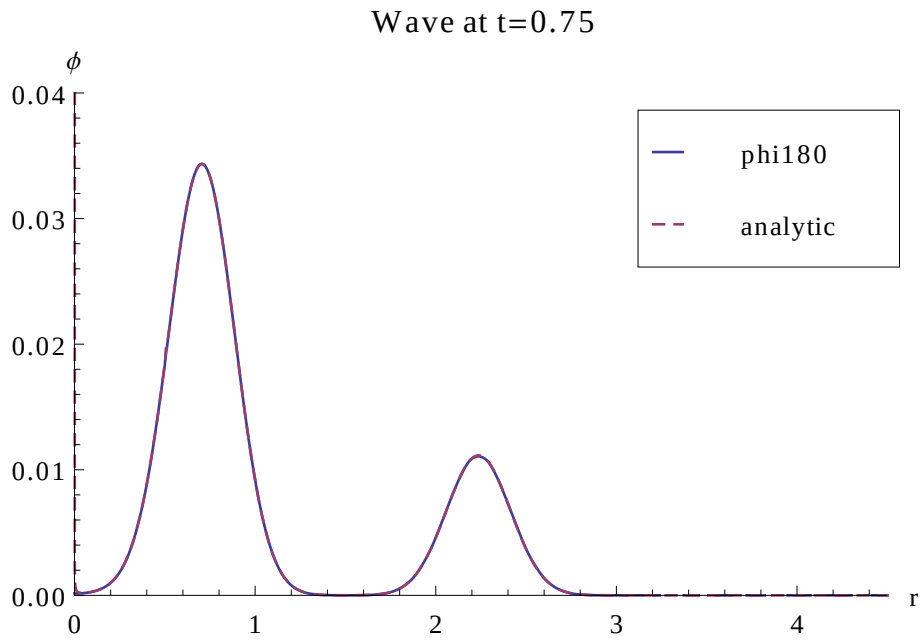


Figure 5.4: The function 'phi80' shows the interpolated simulation results and 'analytic' refers to the analytic solution.

Then we can show the errors for different choices of the finite difference grid resolution in the following logarithmic plot.

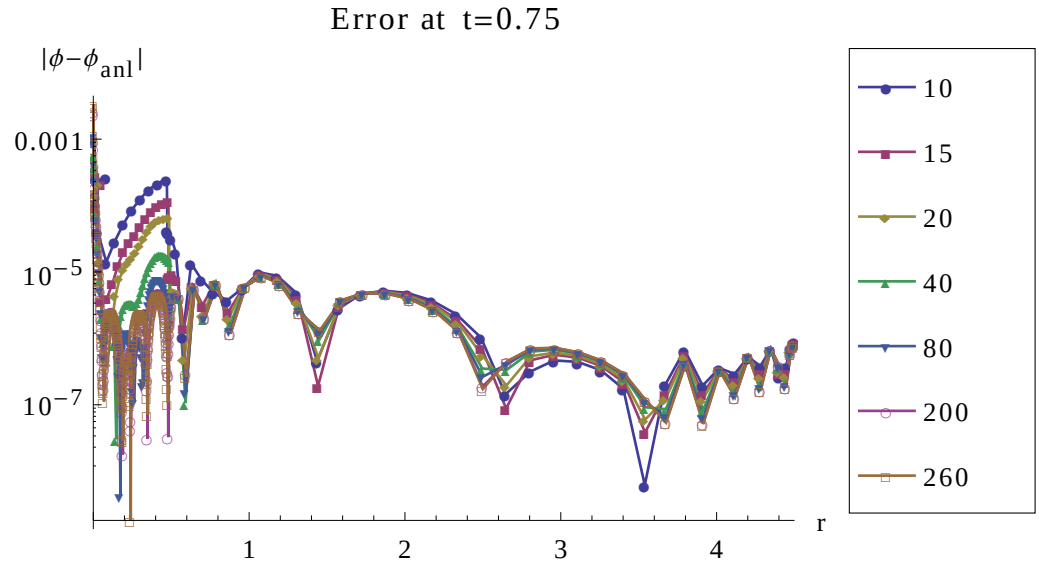


Figure 5.5: Errors for different FD grid resolutions for fixed PS order 40.

We see in the error plot that the errors in the finite difference region decrease with increasing resolution until the finite difference grid spacing coincides with the minimum PS

spacing, i.e. $\Delta x_{PS} \approx \Delta x_{FD} \approx 0.00625$. The following plot shows this result more explicitly. The errors at the points near the origin are caused by the inner boundary conditions and are ignored here.

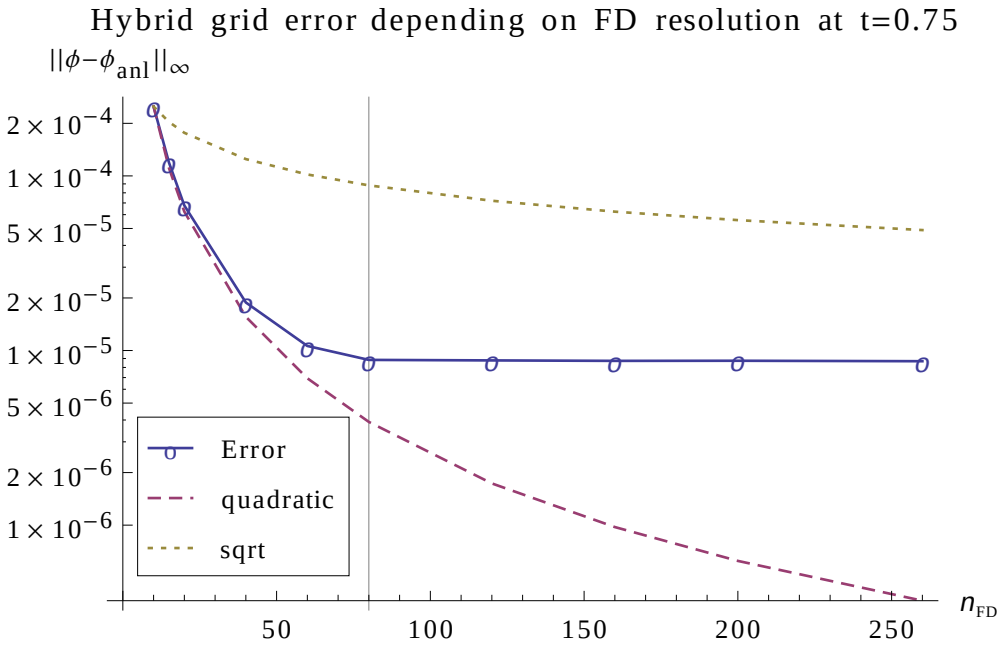


Figure 5.6: Dependency of maximum error on the FD resolution at $t = 0.75$. The vertical line indicates the resolution where at the interface $\Delta x_{FD} \approx \Delta x_{PS}$.

The last plot shows that the errors start decreasing quadratically with the FD resolution. After reaching equal grid spacings at the interface, a further improvement of accuracy is not gained. We conclude that the minimum FD resolution should guarantee equal grid separations at the interface.

In addition, we can compare the results of the final time which was chosen large enough such that the waves have left the domain under consideration. Then the analytical solution is $\phi = 0$ and we take a look at the behavior of the errors there.

Unfortunately we only see the contributions of the outer boundary condition, which dominate the errors caused by the different FD resolutions by far.

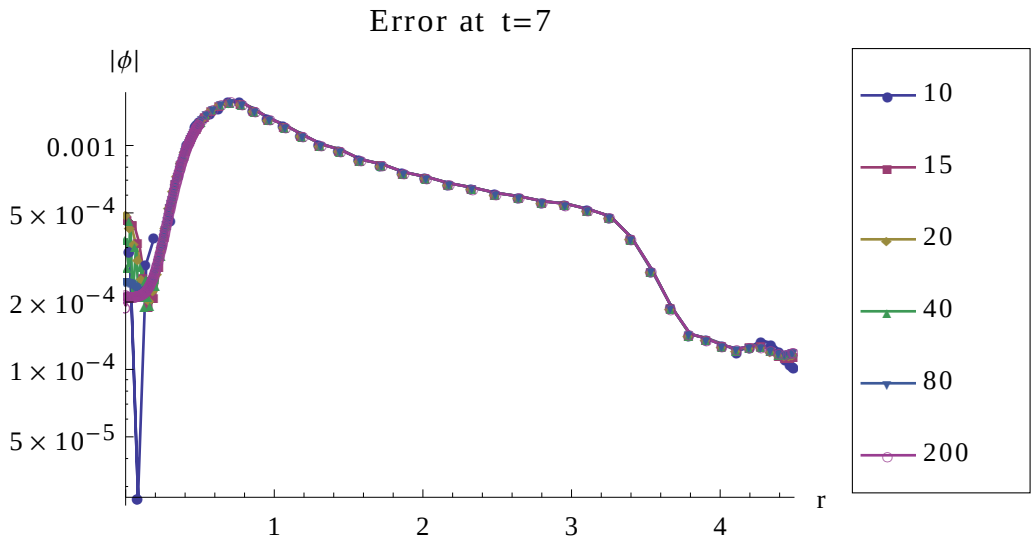


Figure 5.7: Errors for different FD grid resolutions by fixed PS order 40.

5.5.2 Pseudospectral order 60

Analogously to the previous subsection we discuss the dependency of the error on the FD resolution. We now choose a fixed PS order 60. We add some additional FD resolutions, especially 180, which in this configuration leads to equal grid spacings at the interface. Again we first show the simulation results for equal grid spacing and the analytic solution in the following plot:

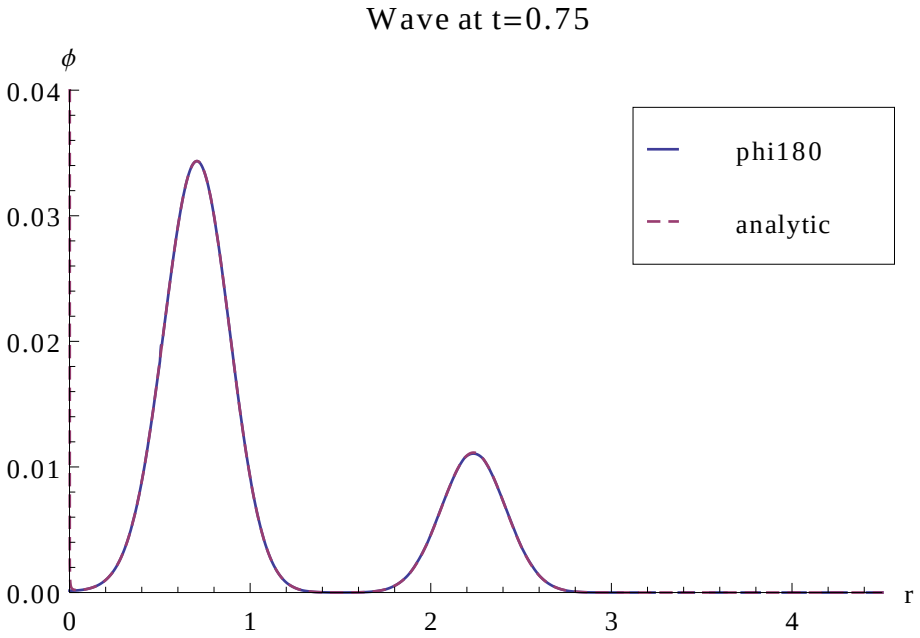


Figure 5.8: The function 'phi180' shows the interpolated simulation results and 'analytic' refers to the analytic solution.

The errors are displayed in the following:

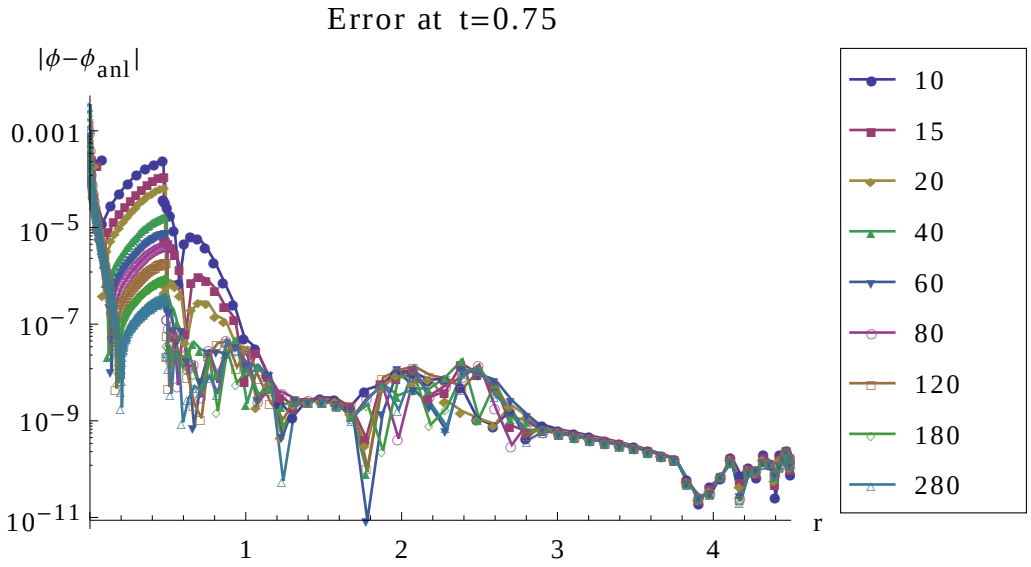


Figure 5.9: Errors for different FD grid resolutions for fixed PS order 60.

By ignoring the points near the inner boundary we can show again that the FD errors decrease nearly quadratically. But now the curve flattens after the resolution with equal grid spacing at the interface.

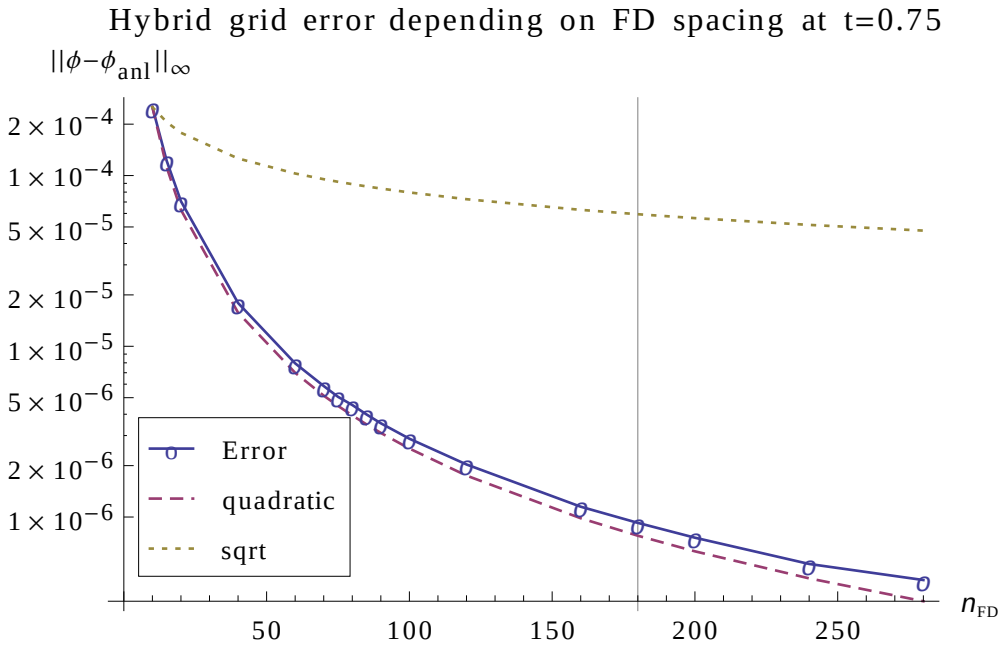


Figure 5.10: Dependency of maximum error on the FD resolution at $t = 0.75$. The vertical line indicates the resolution where where at the interface $dx_{FD} \approx dx_{PS}$.

We conclude that choosing smaller FD grid spacings than the minimum PS grid spacing leads to minimal errors in this situation. After the run with 280 points the time step becomes too large. Nevertheless we expect again a flattening of the curve as in the simulations with PS order 40.

Next we consider the last time slice at $t = 7$. Here we get a new effect compared to the setup in Section 5.5.1.

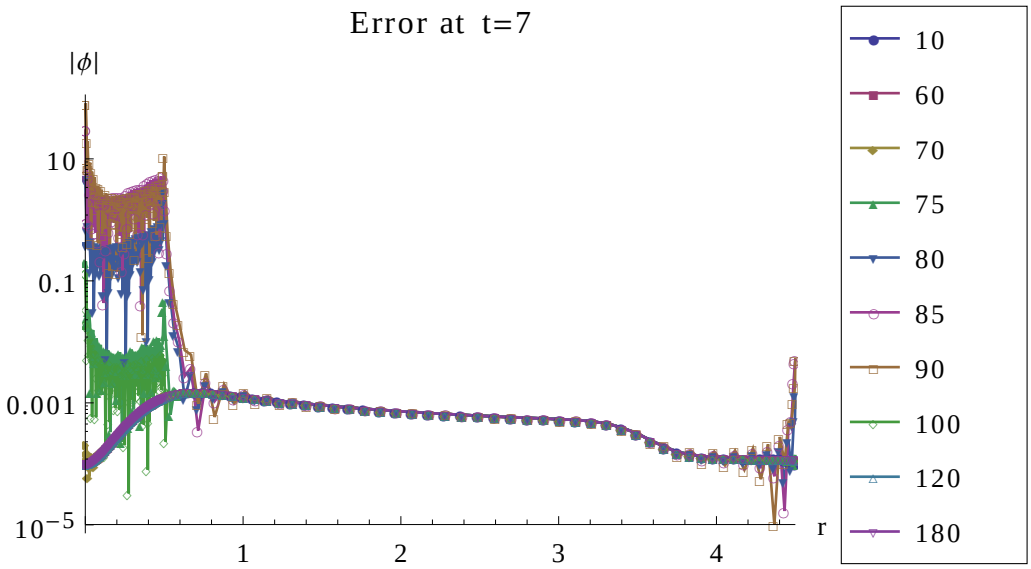


Figure 5.11: Errors for different FD grid resolutions by fixed PS order 60.

At FD resolution 180 the grid separations at the interface coincide. Here and for higher resolutions the FD errors are dominated by the errors of the outer boundary as expected. But now there are simulation runs near to 90 FD points which lead to much higher errors than the other ones. This effect can be seen in the following plot:

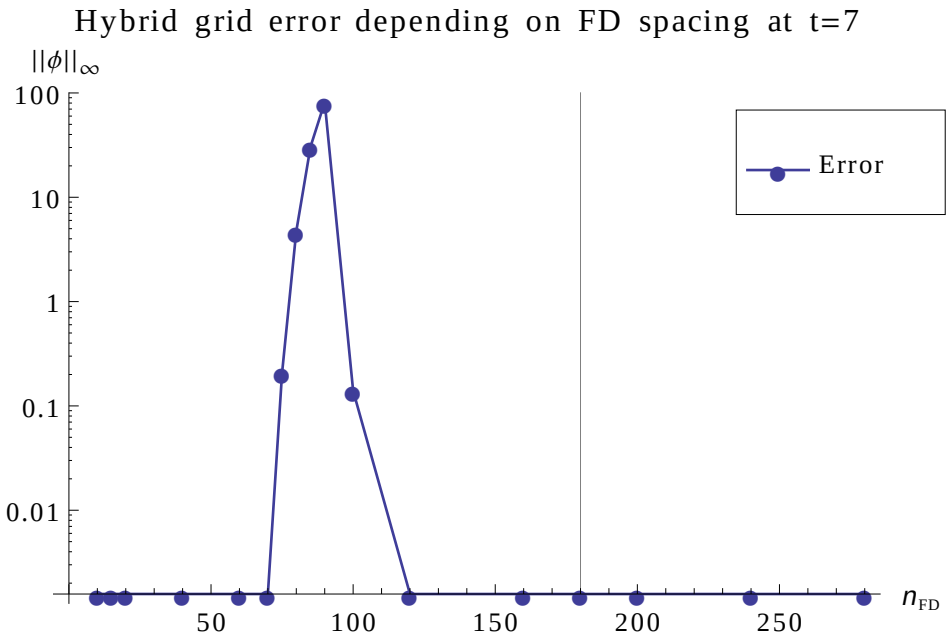


Figure 5.12: Dependency of maximum error on the FD resolution at $t = 7$. The vertical line indicates the resolution where at the interface $dx_{FD} \approx dx_{PS}$.

There seems to be no obvious reason for this behavior, especially because the simulations run well for lower FD resolutions. So up till now, the only interpretation is that some resonances occur when the FD spacings are near to twice the minimum PS spacing. But this was not observed in the case of PS order equal to 40.

We conclude again, that the choice of the FD resolutions should be near to equal grid spacings at the interface or slightly higher to get the optimum accuracy with minimum computation costs.

5.6 Convergence for constant grid-spacing-ratios

Next we consider the situation where we keep the ratio between the FD grid spacing and the minimum PS grid spacing constant, but vary the resolutions.

5.6.1 FD grid spacing equal to minimum PS spacing

The initial data and the grid configuration do not change compared to previous considerations. The time step is set to $dt = 0.0005$. The resolutions of the grids are given in the following table:

FD pts.	PS orders
$\{32, 45, 62, 80, 103, 125, 153, 180, 250, 285\}$	$\{25, 30, 35, 40, 45, 50, 55, 60, 70, 75\}$

Table 5.4: Simulation parameters for constant ration 1 of the grid spacings.

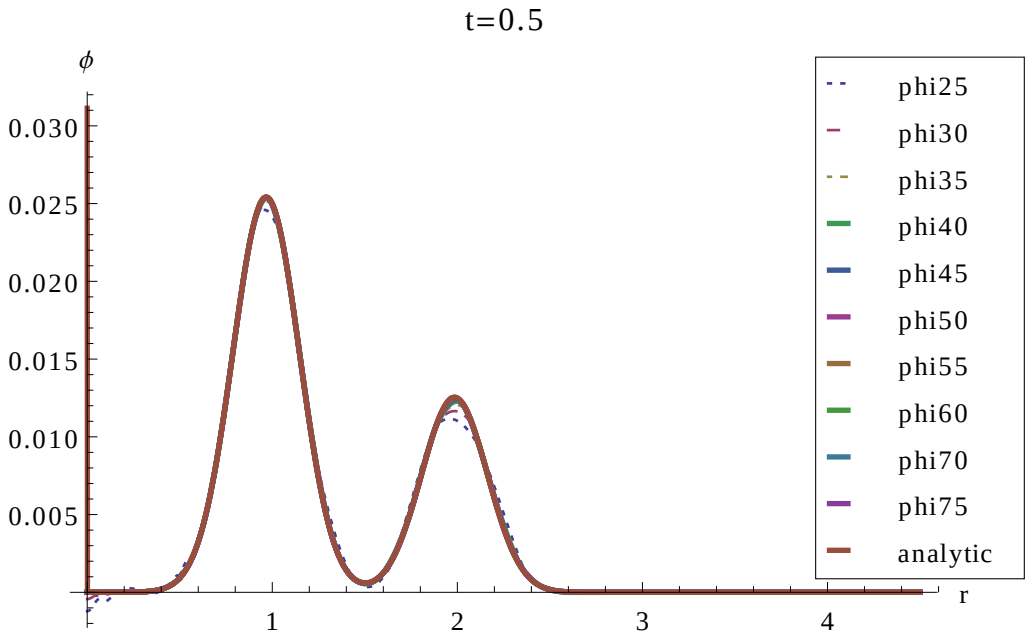


Figure 5.13: Simulation results at $t = 0.5$. The numbers indicate the PS order of the simulation.

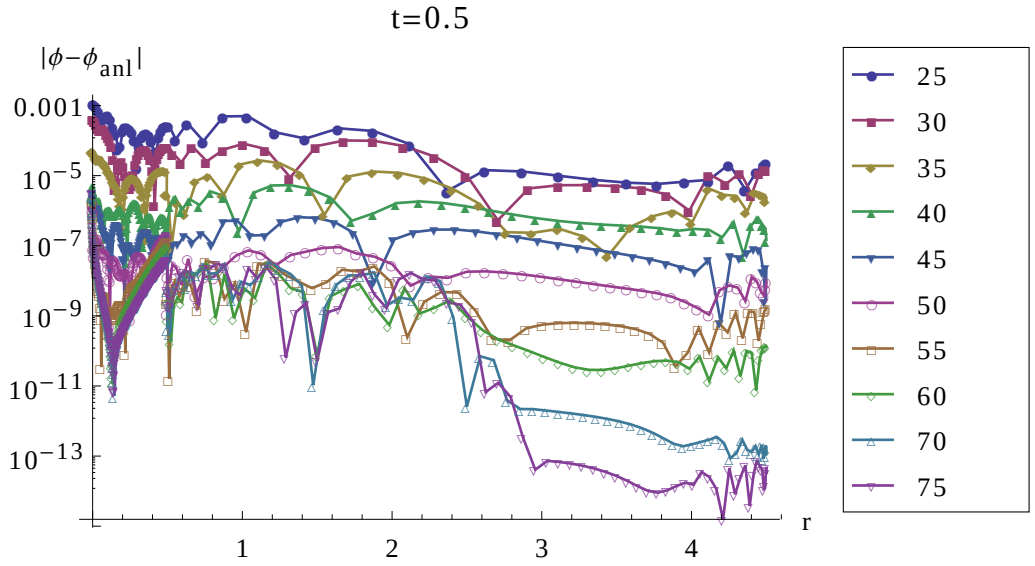


Figure 5.14: Errors on the interval at $t = 0.5$.

The logarithmic plot shows the behavior of the errors compared to the analytic solution

on the simulation interval. We observe nearly spectral decrease of the errors in the outer part of the PS segment, whereas in the inner part the errors of the FD grid dominate. We see in the following plot, how the maximum error on the interval behaves with increasing resolution. Again we ignore the errors near to the inner boundary (approximately the first 5% of the FD interval), which are caused by the inner boundary conditions and the fact that the analytic solution does not implement those conditions.

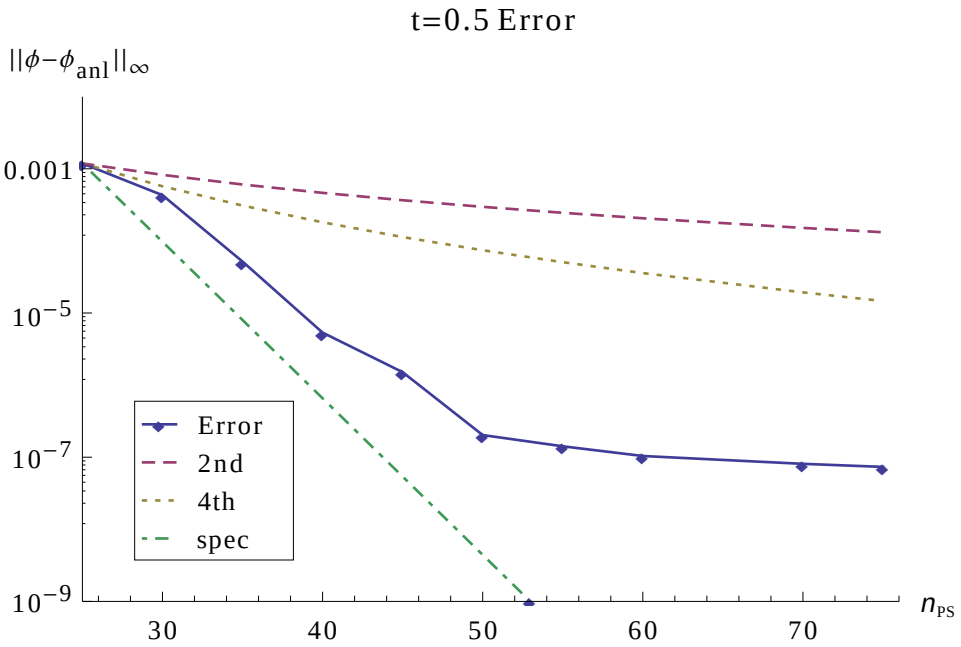


Figure 5.15: Error dependency on the approximation order. For comparison reasons the second, fourth and spectral order plots are shown. For the spectral convergence rate the function $\exp(n/2)$ was chosen.

We observe that the curve flattens near 10^{-7} which corresponds to the errors in the FD grid near the interface.

Except for the two lowest resolutions, the contributions from the outer boundary (note that they increase on their way in with a r^{-1} dependency) dominate the errors of the rest of the simulations in the final time slice. This can be seen in the following plot:

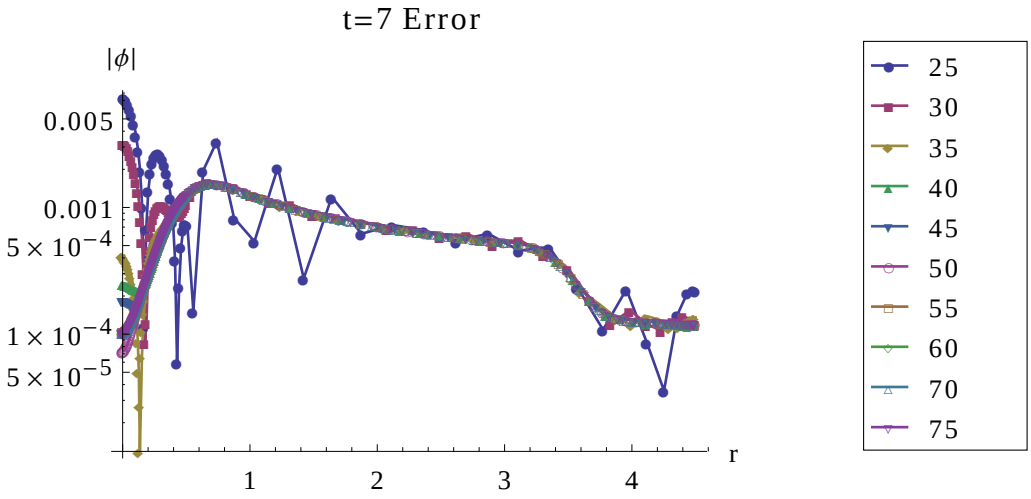


Figure 5.16: Errors on the interval at $t = 7$.

5.6.2 FD grid spacing equal to half of minimal PS spacing

Next we increase the FD resolution by an factor of 2 and choose as time step $dt = 0.00025$. Then our simulation parameters read:

FD pts.	PS orders
$\{64, 90, 124, 160, 206, 250, 306, 360, 428, 500, 570\}$	$\{25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75\}$

Table 5.5: Simulation parameters for FD grid spacing equal to half of minimum PS spacing.

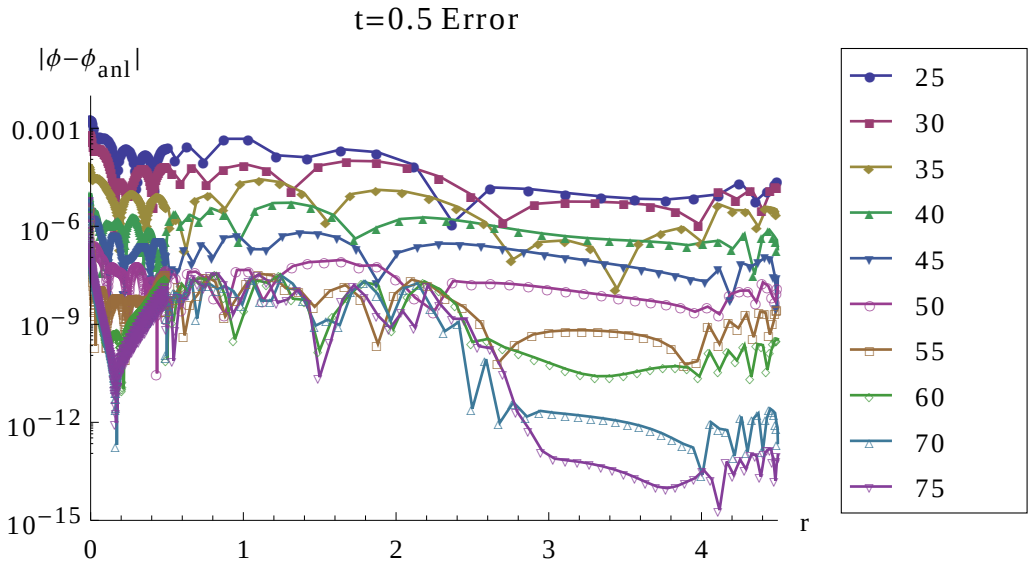


Figure 5.17: Errors on the interval at $t = 0.5$.

The comparison of the maximum errors per resolution to the previous runs with equal grid spacing show only a slight improvement. So we conclude, that refining only the FD resolution has an effect on the errors, but this effect is very small and not worth the large additional computation costs.

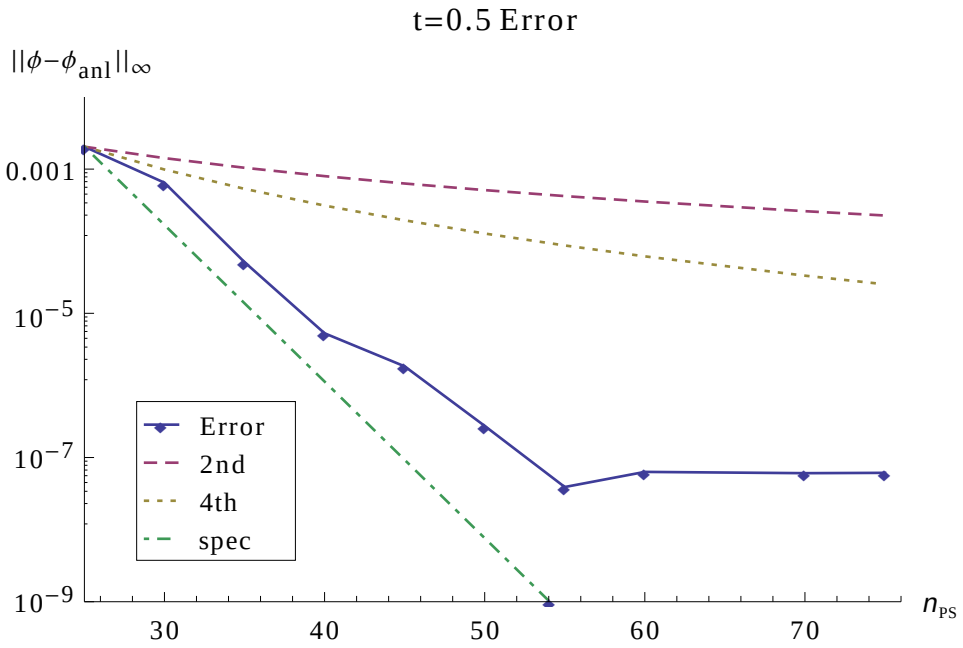


Figure 5.18: Error dependency on the approximation order. For comparison second, fourth and spectral order is shown. For the comparison with a spectral convergence rate the function $\exp(n/2)$ was chosen.

The results we get for the last time slice are similar to the ones in the previous subsection.

5.7 Conclusion

- In this chapter we have discussed another toy model, namely the wave equation in spherical symmetry. Implementing the hybrid grid method for this equation improves the insight into problems which will also arise for the GBSSN evolution system. The major change is the introduction of the cell centered finite difference grid to handle singularities at the inner boundary.

- Simulations have shown increased errors near the origin. This might lead to oscillations during longer runs. We will have to introduce filtering methods in the GBSSN simulations.

- In addition, we observe that it is very important to identify the interface points correctly. In case of the cell centered grid, it is not enough to keep the last FD point and

the first PS point separated by a half of the grid spacing. They really have to coincide to get the best results.

- We also saw that increasing the FD grid spacing might cause problems, whereas decreasing them does not lead to much better results. So we will aim at approximately equal grid spacings at the interface or slightly smaller FD spacings.
- For lower resolutions we observe a spectral convergence rate in the hybrid grid. This curve flattens when the FD errors start to dominate.

6 GBSSN evolution

This chapter covers the results of the GBSSN system with wormhole and trumpet initial data which are derived from the Schwarzschild metric.

6.1 Wormhole puncture initial data

In this section we will derive the initial data for wormhole puncture simulations from the well known expressions for the Schwarzschild metric.

The line element of the Schwarzschild metric in Schwarzschild coordinates is given by:

$$ds^2 = -\left(1 - \frac{2M}{R}\right) dT^2 + \left(\frac{1}{1 - \frac{2M}{R}}\right) dR^2 + R^2 \left(d\theta^2 + \sin^2(\theta)\ d\phi^2\right) \tag{6.1}$$

Following [20, p. 4] we modify this expression by imposing the coordinate transformation

$$R = \chi^{-\frac{1}{2}} r \quad \text{with} \quad \chi = \left(1 + \frac{M}{2r}\right)^{-4}. \tag{6.2}$$

Then Equation 6.1 becomes:

$$ds^2 = -\left(\frac{1 - \frac{M}{2r}}{1 + \frac{M}{2r}}\right) dT^2 + \chi^{-1} \left(dr^2 + r^2 \left(d\theta^2 + \sin^2(\theta)\ d\phi^2\right)\right) \tag{6.3}$$

We will refer to this expression as the *Schwarzschild metric in isotropic coordinates*. The $T = \text{const}$ slices are topologically $\mathbb{R}_+ \times \mathbb{S}^2$ and by setting $\theta = \frac{\pi}{2}$ we arrive at the well known *wormhole* picture. The singularity at $R = 0$ is not reached by the isotropic radius r. Instead $R \to \infty$ for large and small r. The minimum $R = 2M$, which is equal to the *event horizon*, is reached for $r = \frac{M}{2}$. So we have now two copies of the space outside of the event horizon. They are connected by a wormhole with its throat at $R = 2M$.

This is the basic description of initial data for puncture black hole simulations with a single non-spinning black hole. We will refer to the point $r = 0$, which represents the second asymptotically flat end, as the *puncture*.

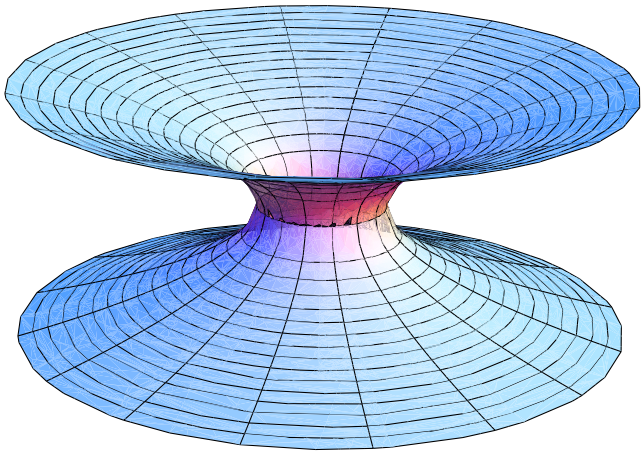


Figure 6.1: Wormhole picture: Schwarzschild metric in isotropic coordinates with $T = \text{const}$ and $\theta = \frac{\pi}{2}$. The upper and lower part of this picture lie in two different universes. The throat of the wormhole represents the Schwarzschild radius.

The initial values for most variables are guided by practical assumptions. We use the Minkowski metric for the background metric terms in Equation 2.83. Then we get as initial data for the conformal factor

$$\chi = \left(1 + \frac{M}{2r}\right)^{-4} \tag{6.4}$$

and we use the *pre-collapsed lapse* condition for α:

$$\alpha = \left(1 + \frac{M}{2r}\right)^{-2} \tag{6.5}$$

This leads to $g_{rr} = 1$ and $g_{\theta\theta} = 1$. We consider vanishing shift β^r at the initial time slice. These choices imply that the component A_{rr} of the extrinsic curvature and its trace K vanish. The same is true for the background connection difference variable Λ^r and the parameter vector field for the Gamma-driver shift B_r.

6.2 Trumpet initial data

In the previous section we discussed a straightforward construction of initial data for the GBSSN evolution. While we could evolve the initial data described above with trivial gauge conditions, we are interested in performing evolutions with gauge conditions similar to the ones chosen for binary black hole puncture evolutions (see Section 2.4).

We want to construct initial data with a known, time independent solution. This is outlined in [20, p. 4] and will allow us to track the errors, which arise through the evolution, in a direct manner with non-trivial gauge conditions.

First of all it is shown in [20, p. 4] that for spacetimes with two asymptotically flat ends, maximal slicing can not lead to time independent results and we have to find an alternative

method. The equations for a maximal slice of the Schwarzschild metric are given by:

$$g_{RR} = \left(1 - \frac{2M}{R} + \frac{C^2}{R^4}\right)^{-1} \tag{6.6a}$$

$$K_i{}^j = \text{diag}\left(-\frac{2C}{R^3}, \frac{C}{R^3}, \frac{C}{R^3}\right) \tag{6.6b}$$

$$\beta^r = \frac{\alpha C}{R^2} \tag{6.6c}$$

$$\alpha = \sqrt{1 - \frac{2M}{R} + \frac{C^2}{R^4}} \tag{6.6d}$$

Here $C = \frac{3\sqrt{3}M^2}{4}$ and $R \in [1.5M, \infty)$, which leads to $\alpha \geq 0$ in the whole domain and $\alpha = 0$ and $\beta^r = 0$ at $R = 1.5M$. Next we observe that the spatial metric diverges at $R_0 = 1.5M$. This means that each point of the interval $(1.5M, \infty)$ has infinite distance to $R_0 = 1.5M$. If we approach this radius value from inside the interval, the time slice becomes a cylinder with radius $R_0 = 1.5M$. As discussed in [20, p. 5], this situation is referred to as *trumpet data*. For appropriate choices of lapse and shift, this data become time independent.

For the lapse one can use the following trick: Instead of solving an elliptic equation in each time step to find the right expression satisfying the maximal slicing condition, one can evolve the lapse with a special form of the $1 + \log$-condition, as was already discussed in Equation 2.87. If K is equal to zero, the lapse will not evolve and maximal slicing is maintained.

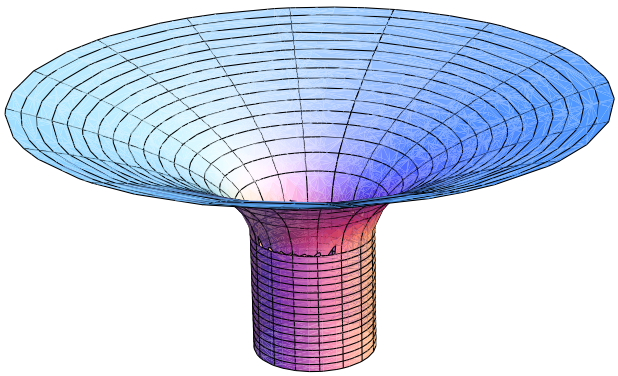


Figure 6.2: Trumpet picture: Trumpet data with $T = \text{const}$ and $\theta = \frac{\pi}{2}$. The upper part describes the asymptotically flat spacetime. Then the infinitely long cylinder starts at $R_0 = 1.5M$.

To use it as initial data for our simulations, we first have to transform from Schwarzschild coordinates to isotropic coordinates. This can be done analytically as described in [3]. After imposing the required coordinate transformations we get an implicit relation for the lapse α. This can be solved numerically on an predefined grid and used for the calculation of the initial data of χ, A_{rr} and β^r. The other variables keep the simple form they had before: $g_{rr} = 1$, $g_{\theta\theta} = 1$, $K = 0$, $\Lambda^r = 0$ and $B_r = 0$.

As pointed out in [20, p. 10], simulations starting with wormhole puncture initial data approach after some unit time intervals (for $M = 1$) the trumpet picture. So starting already with trumpet data skips this transition phase and may reduce errors during the simulation.

6.3 Inner boundary conditions

Several ways of treating the inner boundary, i.e. the center with coordinate radius $r = 0$, were tested throughout the simulation process. In the following we discuss briefly the most promising ones. We use a cell centered finite difference grid next to the origin, as the equations contain singular terms at the origin.

6.3.1 Symmetry/antisymmetry conditions

We consider a system in spherical symmetry. Therefore the solution has to satisfy some symmetry or antisymmetry conditions at the origin. These conditions can be incorporated by the introduction of *ghost points* with negative values of the radius. Their number depends on the order of the finite difference grid. A second order FD method requires only one ghost point, a fourth order method already two. From the GBSSN system it follows that the variables χ, $g_{\theta\theta}$, g_{rr}, A_{rr}, K and α are symmetric at $r = 0$, whereas Λ^r, β^r and B_r are antisymmetric.

The ghost points are set to the equal value of their counterpart with positive radius for the symmetric fields, and to the negative value for the antisymmetric ones.

6.3.2 Setting β^r and $g_{\theta\theta}$ at innermost points

This condition follows the principle of trial and error and is a minimalistic approach to the problem. We do not introduce ghost points, so the innermost point of the FD grid still has a positive radius r. Depending on the finite difference order, derivatives at the first (FD second order) or the first and the second (FD fourth order) point next to the origin are calculated by left sided stencils, which were discussed in Section 1.2.

Only the values for β^r and $g_{\theta\theta}$ are set via a higher order Taylor polynomial. The calculation of this polynomial incorporates the values $\beta^r(0) = 0$ and $g_{\theta\theta}(0) = 1$ (we extracted a factor r^2 compared to the terms in the literature) at the origin. This method was suggested in [8, p. 5] for wormhole puncture initial data.

6.3.3 Stationary trumpet data

For simulations with trumpet initial data, where we treat a stationary solution, we can just save the initial values at the innermost points and use them as fixed boundary values. Unfortunately, in longer simulation runs this approach leads to instabilities. Even if we set

only some variables and let the rest undergo free evolution, this approach does not seem to give stable results.

6.4 The hybrid grid and the interface condition

As described previously for the wave equation in Section 4.1.2, we want to construct a hybrid grid for the GBSSN system in spherical symmetry and define appropriate interface conditions.

In spherical symmetry we have to discretize one space dimension, i.e. we use the radial coordinate only. We place a cell centered finite difference grid next to the origin. Then we append a single pseudospectral grid for our tests. The radius where these two grid intervals meet is called the *interface radius* r_{int}.

First we should mention, that the most naive way of setting the interface conditions, namely taking just the arithmetic average of the values at the interface points, led only to instabilities already at times between $t = 1M$ and $t = 2M$.

One important observation we made through the test process is the following: The last point of the finite difference grid and the first point of the pseudospectral grid have to be identified. Even if we use a cell centered finite difference grid, it is not enough to keep the last point of the FD grid separated from the first PS grid point by only a half grid interval. This would already lead to large additional errors during long time runs.

The next step is the formulation of the interface conditions. We require the characteristic fields stated in Table 3.6. Afterwards we calculate the values of these fields, first at the last grid point of the finite difference interval and then at the first grid point of the pseudospectral grid interval. We will denote those values in the following by

$$X_i[\text{l}] \quad \text{and} \quad X_i[\text{r}]. \tag{6.7}$$

Furthermore we have to calculate the eigenvalues of the system at the interface position. For that we use Table 3.4. Again this is done for the last FD point and the first PS point and we get two sets of eigenvalues denoted by

$$\lambda_i[\text{l}] \quad \text{and} \quad \lambda_i[\text{r}]. \tag{6.8}$$

Of course, in the optimal case these two sets coincide, but through numerical errors they may differ by a small value.

After the calculation of these expressions, we check the sign of the eigenvalue pairs. In the very unlikely case, where the signs of the eigenvalues differ at the interface point, we have to choose one so that the simulation can continue without throwing an exception. The easiest example when this may happen is a nearly vanishing value of β^r, which changes the sign at the interface position.

Those eigenvalues are by Section 3.1 the speeds of the characteristic fields. Therefore, in

spherical symmetry their signs tell us if they are outgoing (positive) or incoming (negative). This brings us to the key idea of the interface formulation: For the further calculations we use the values X_i[l] if the fields are outgoing and we use X_i[r] if they are incoming.

In case of the field X_1 with $\lambda_1 = 0$ we have to choose. We can not ignore it, because we require its value for further calculations. We assume that the pseudospectral grid is of higher accuracy, so we use the value of the outer interval. Nevertheless, there is no rigorous argument leading to this choice. Both possibilities did not show any significant difference in the outcome of our simulations.

After these considerations only one list with nine elements represents the values of the characteristic fields at the interface. We denote it by

$$X_i[\text{if}]. \tag{6.9}$$

In Table 3.7 we stated the formulas required for the calculation of the standard GBSSN variables by given values of the characteristic fields. As the reader may have already observed, they depend on the values of χ, g_{rr}, $g_{\theta\theta}$, α and β^r. Furthermore, the formulas give only values for the spatial derivatives of these variables: Q_χ, $Q_{g_{rr}}$, $Q_{g_{\theta\theta}}$, Q_α and Q_{β^r}. So the following questions arise: Which values should be inserted for χ, g_{rr}, $g_{\theta\theta}$, α and β^r throughout the calculation and how can we calculate the interface values of those variables.

The simplest (and also chosen) solution to the first problem is the following: We have two lists with these variables from our interface, one from the last FD point and one from the first PS point. We use for the calculation the arithmetic average of these values.

The second problem is more serious. From our analysis we get only the values of the derivatives. The question arises, how we can calculate the required values from this information. The most trivial approach was leaving the values of these variables unchanged, i.e. using the one sided approximations calculated by the differentiation schemes. This worked surprisingly well for surprisingly long times. At some stage the errors grew larger and the difference between the last FD value and the first PS values grew in time.

A much better approach is to incorporate the derivative values by using the Simpson formula for the integration:

$$\int_a^b Q(r)dr = \frac{b-a}{6}\left(Q(a) + 4\cdot Q\left(\frac{a+b}{2}\right) + Q(b)\right) + \mathrm{O}\left(\left(\frac{b-a}{2}\right)^4\right)$$

The last term indicates the fourth order accuracy of this quadrature rule. We integrate over the last three points of the FD grid and get the interface values for χ, g_{rr}, $g_{\theta\theta}$, α and β^r. Finally we have to set the so calculated values for the last FD and first PS point, which finishes the calculation at the interface. We summarize the algorithm for the computation of the interface values in Fig. 6.4.

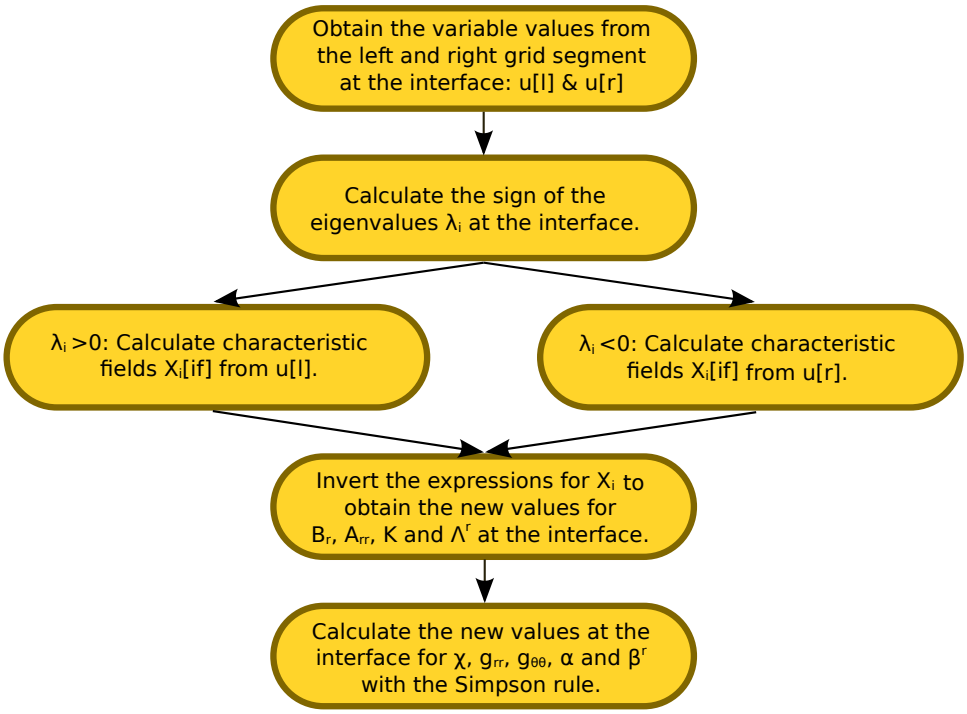


Figure 6.3: Illustration of the algorithm to determine the interface conditions.

6.5 Outer boundary conditions

In principle we use two versions of outer boundary conditions: A radiation condition and stationary data for the trumpet case. In general we observe, that increasing the outer radius of the physical interval has always positive effects on the stability of the outer boundary conditions.

6.5.1 Radiation boundary condition

This condition is constructed analogously to Section 4.1.1. We calculate the eigenvalues and the values of the characteristic fields given by Table 3.4 and Table 3.6 at the outermost point of our physical interval.

Next we set all the characteristic field values of the incoming fields (eigenvalues smaller than zero) to zero. With this set of variables and Table 3.7 we calculate the new boundary values. Here we have to face the same problems as the ones described in Section 6.4. This time we have only one set of values for χ, g_{rr}, $g_{\theta\theta}$, α and β^r available, so we use it. Again we can only set values for A_{rr}, K, Λ^r and B_r and have to drop the rest of the information.

6.5.2 Stationary trumpet data

In case of the trumpet evolution we know already the boundary values for all the simulation time, so we can just set it in each times step. Contrary to the same approach for the inner

boundary, at the outer boundary this works pretty well.

6.6 Wormhole puncture evolution without filtering

The configuration in this section is similar to the one discussed in [8]. In contrast to the single interval finite difference method used there, we consider a hybrid grid.

6.6.1 Convergence test in variable α

We choose a fixed time step size for all resolutions and radiation boundary conditions at the outer boundary. At the inner boundary we implemented conditions for β^r and $g_{\theta\theta}$ only, as it was described in Section 6.3.2. We use a fourth order finite difference method.

Additional parameter choices are given in the following table, where the parameter η denotes the constant in the Gamma-driver shift condition Equation 2.92.:

dt	phys. domain FD	phys. domain PS	FD pts.	PS orders	η
0.005	$[0, 10]$	$[10, 50]$	$\{257, 321, 401\}$	$\{35, 40, 45\}$	6

Table 6.1: Simulation parameters for convergence tests.

We start by showing a logarithmic error plot for the finite difference grid segment at $t = 2.5M$ for the α function. The third color in the plot is the difference of high and medium resolution runs, multiplied by the expected convergence factor calculated for the finite difference point numbers under the assumption of a fourth order method.

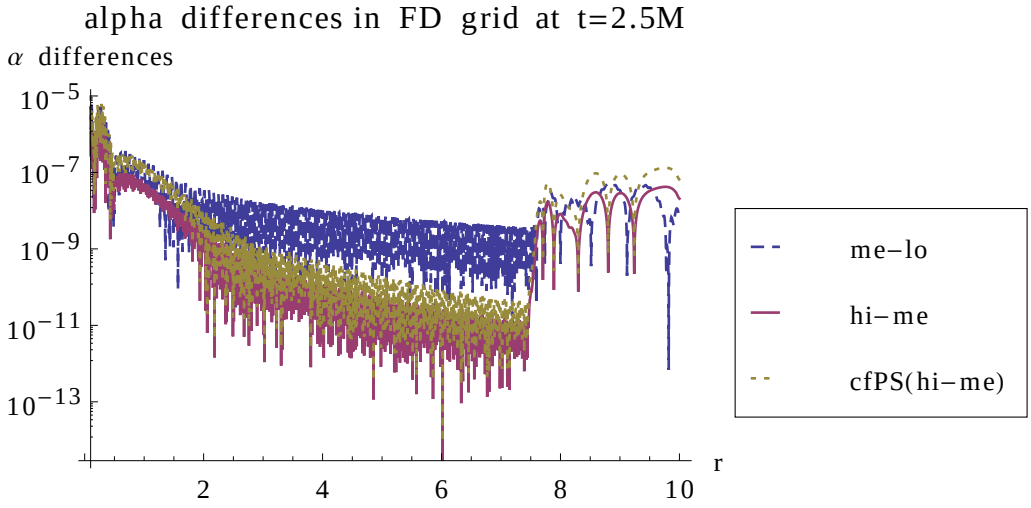


Figure 6.4: Convergence test of the finite difference grid segment at $t = 2.5M$.

We see in the center of the grid segment a much higher order of convergence than expected.

But the errors induced by the the inner boundary and the interface already started to destroy that result. After $t = 10M$ (one light crossing time of the FD interval) the error contributions, which originated at the boundaries of the FD interval, dominate. With that in mind, we consider the whole physical interval at $t = 50M$. The third plot shows the expected values with spectral convergence factor calculated from the pseudospectral orders.

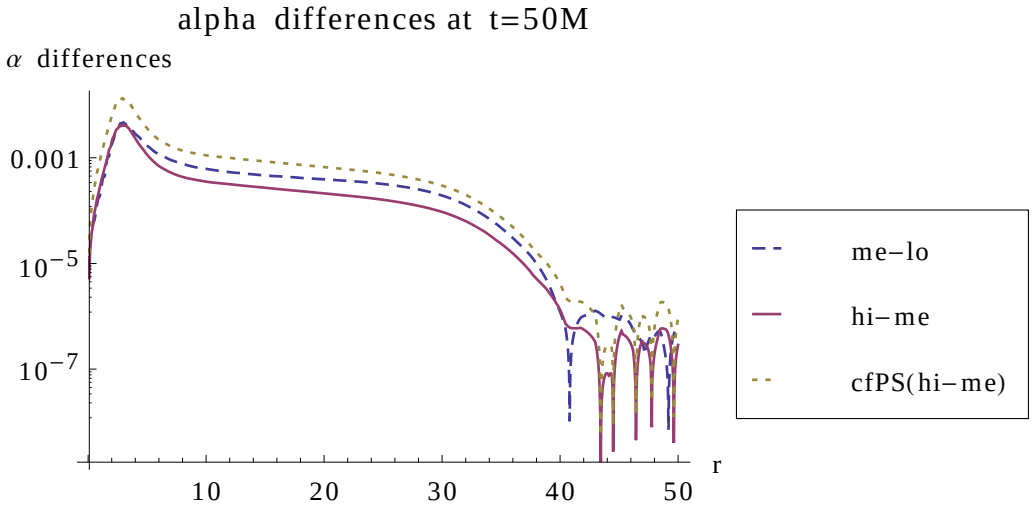


Figure 6.5: Convergence test of the whole physical interval at $t = 50M$.

In general the errors increased during the additional elapse of time. The finite difference part does not show convergence anymore, but the pseudospectral part does. The convergence rate is less than expected for a pseudospectral simulation. The reason is the finite difference grid contributing to the larger errors. Also the radiation boundary conditions contribute, but not as much as one would expect.

6.6.2 Constraints

Here we show plots of the Hamiltonian and C^r constraints calculated from the simulation data. The momentum constraint gives a similar result as the Hamiltonian one. Each plot contains the data of all three resolutions.

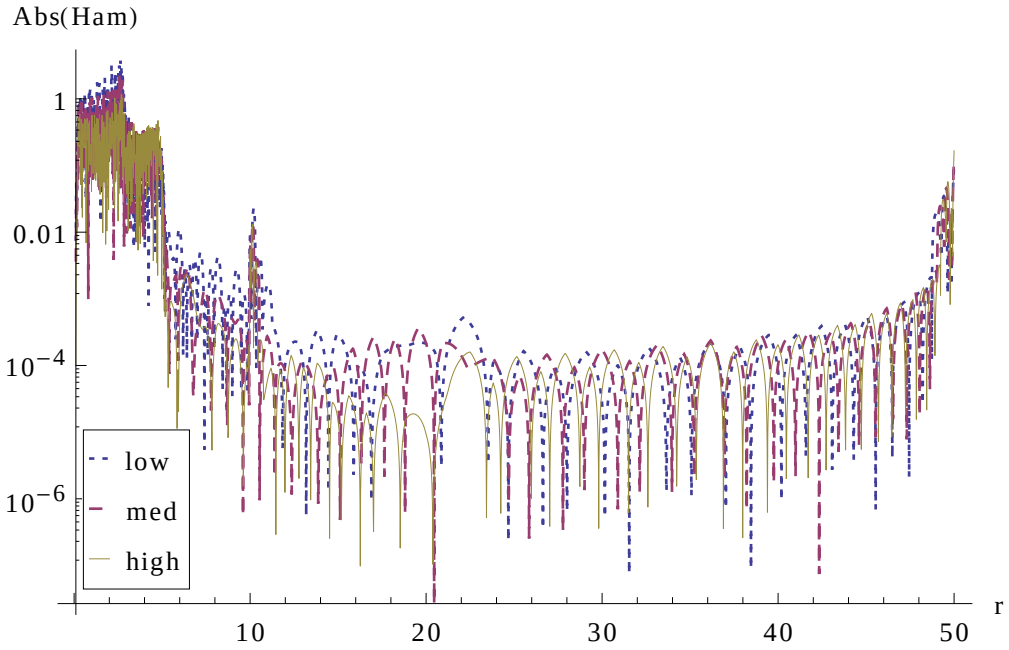


Figure 6.6: Absolute value of the Hamiltonian constraint at $t = 50M$. The dotted line shows the lowest resolution, the dashed line the medium resolution and the thin line the highest resolution result.

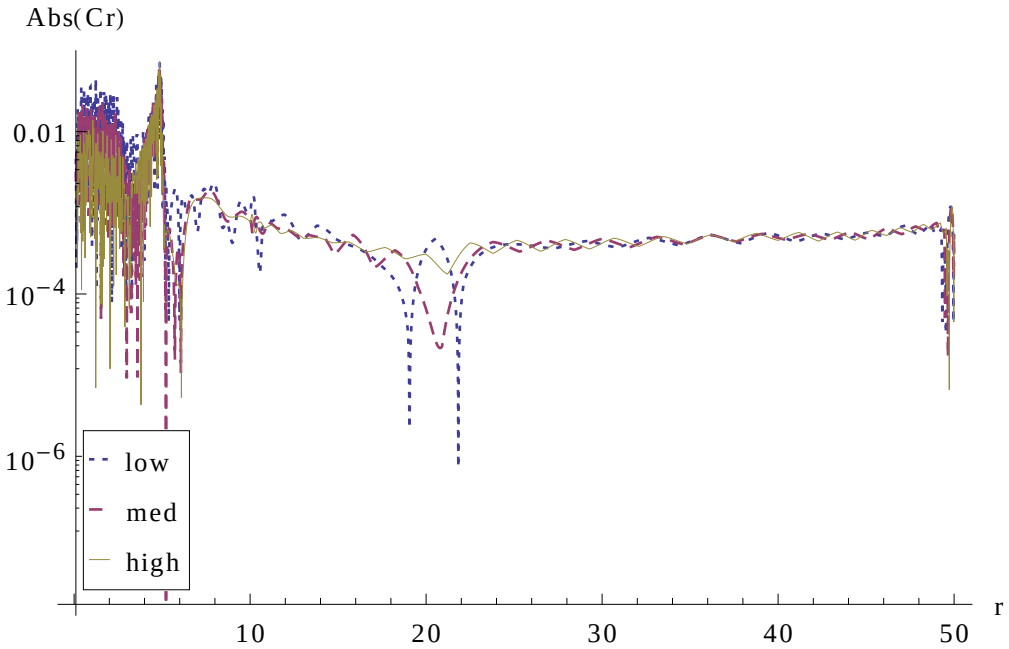


Figure 6.7: Absolute value of the C^r constraint at $t = 50M$. The dotted line shows the lowest resolution, the dashed line the medium resolution and the thin line the highest resolution result.

6.6.3 Constraints convergence test

Covergence for constraints leads to similar results. While we get nice convergence in the FD grid for early times, the errors of the PS grid which get transmitted by the interface destroy it. In later times the FD grid gives the largest error contributions, which start to grow in the center and spread over the rest of the interval.

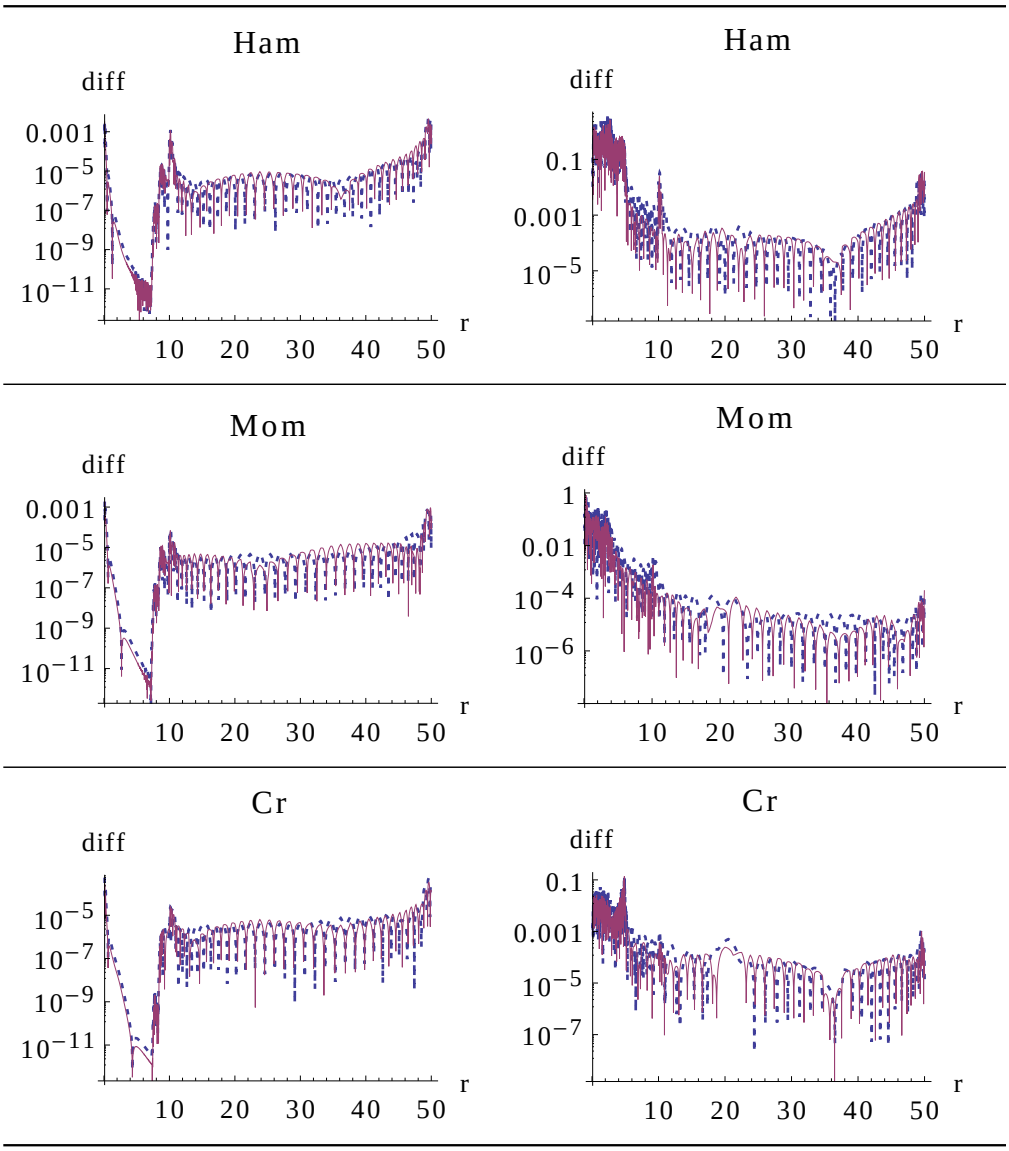


Table 6.2: Constraints at $t = 2.5M$ (first column) and $t = 50M$ (second column). The blue dotted lines always show the absolute value of the difference between the medium and low resolution, the magenta line always shows the absolute value of the difference between the high and medium resolution.

6.6.4 Long run without filtering

In the following we show the results of a wormhole puncture simulation with parameters:

dt	phys. domain FD	phys. domain PS	FD pts.	PS order	η
0.005	$[0, 20]$	$[20, 60]$	801	45	6

Table 6.3: Simulation parameters for $t = 250M$ run.

Compared with the previous runs we extended the FD grid without changing the PS segment length or order.

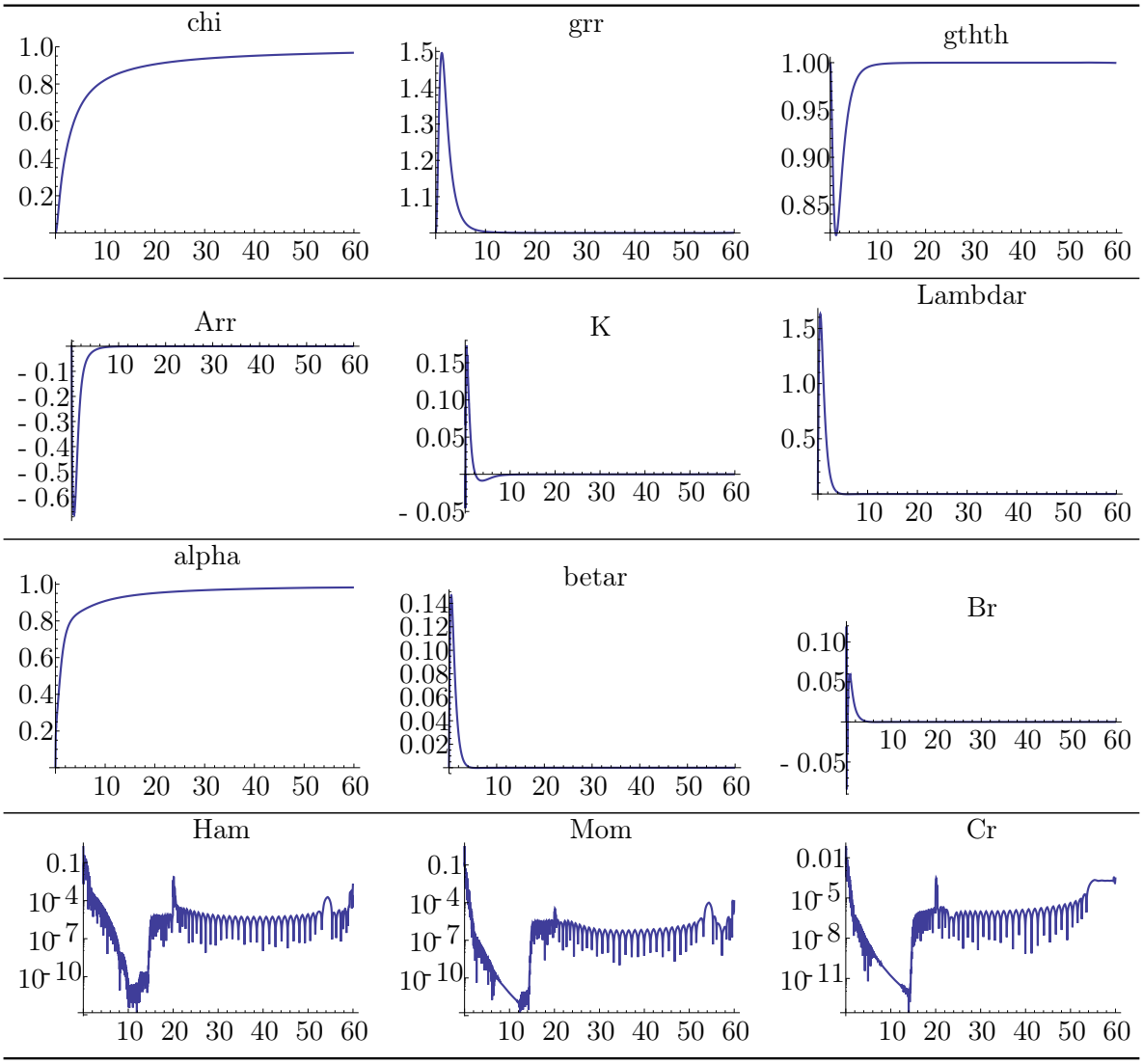


Table 6.4: GBSSN variables at $t = 250M$. The absolute values of the constraints are displayed logarithmically.

The simulation leads to a breakdown either when the errors at the interface grow to large, or the gradients in the GBSSN variables become to large in the FD grid. The last observation was also described in [8].

6.6.5 Conclusion

- We saw in the tests of this section, that the errors in the FD grid lead to instabilities which cause the simulations to crash. These errors are caused by the inappropriate inner boundary conditions and are amplified over time. These considerations lead to the use of filtering techniques for the FD grid.
- We also observe, that increasing the length of the FD grid decreases the errors in the inner part of the grid and improves the run time of the simulation.
- The Gamma driver parameter η also has an influence on the run time. The reason is the inner boundary condition. It seems to work better for higher values of η which also increases the gradients near the inner boundary.

7 GBSSN long time runs

After the discussion of various details about the GBSSN system, initial data and numerical methods, we will now try to produce simulations which run for long times in a stable manner. Hence we have to apply filtering techniques.

7.1 Filtering

We already mentioned filtering in section Section 1.7 and explained the ideas of Kreiss-Oliger dissipation in the FD grid and the exponential filter in the PS grid. After several test runs the introduction of Kreiss-Oliger dissipation led to impressive improvements of long time stability.

Unfortunately the introduction of the exponential filter in the pseudospectral segment did not work out as well. Even though several configurations were tested, the run times decreased compared to the Kreiss-Oliger only case.

This does not mean that no filtering technique for the PS grid is applicable. We stopped trying to implement such methods because the filtering process in the FD grid led already to impressive results, which is described in the following.

In each calculation of the right hand side of the GBSSN system (there are 4 calculations in each Runge-Kutta step), we add an additional term to each right hand side of the variables, which consists of a filtering factor $\Delta x^3 \cdot \epsilon$ and a derivative of the specific variable. In case of the second order GBSSN system we take the fourth derivative.

Theoretical considerations lead to a first estimation of the filtering factor. Nevertheless, only various simulation runs can show, which choice gives the best stability results without too much disturbance to the solution.

It turns out, that filtering the FD segment alone allows run times of several thousand mass-time intervals ($t = 15000M$). So the implementation of a filter for the PS grid was postponed until experiments show the necessity for introducing it.

7.2 Simulation run with very short FD grid and wormhole puncture data

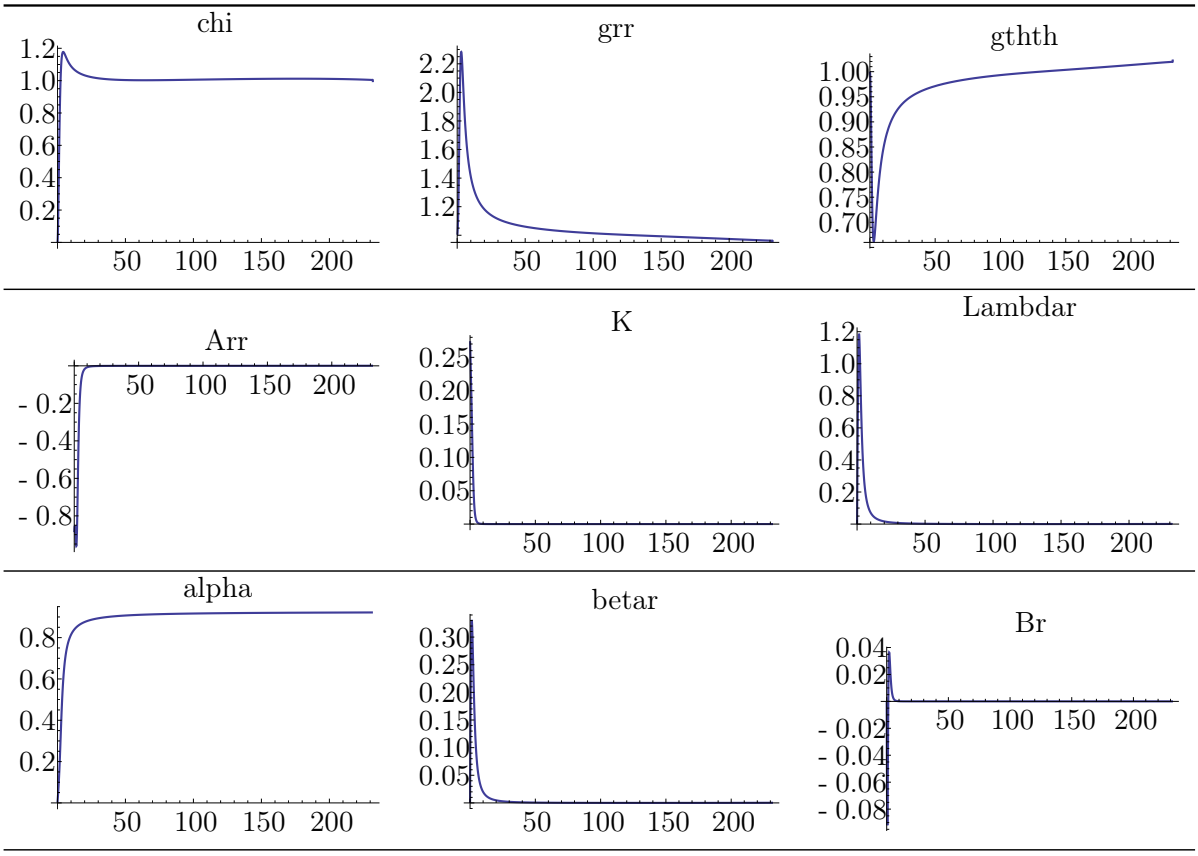
As initial conditions for this test run we use wormhole puncture data with the choice of $\eta = 2$ for the Gamma-driver shift constant. We will use symmetry/antisymmetry conditions for the

inner boundary, as in all simulations in this chapter. The filtering procedure will cancel most of the disturbances introduced with this condition. The reason for the enlarged PS grid is the increased light crossing time for the whole physical interval. It turned out, that this leads to longer run times and better behavior of the radiation boundary conditions. Following the conclusions made in the wave equation chapters, the interface point is exactly identified and we use slightly smaller FD grid spacings than the smallest PS grid spacing. The following table states additional parameters.

dt	ph. dom. FD	ph. dom. PS	FD pts.	PS order	$\Delta x^3 \cdot \epsilon$
0.00125	$[0, 2]$	$[2, 232]$	201	190	0.000001

Table 7.1: Simulation parameters for test run with interface radius $r = 2M$ and outer boundary at $r = 232M$.

In this simulation run we use a short FD grid with interface radius $r = 2M$. This is of special interest due to the fact that the *apparent horizon* (the outermost trapped surface of the black hole) is of this order and lies at roughly half of this value. Even though we observe growing run times with increasing FD grid length (up till $r = 20$), this configuration allows already simulation runs with satisfying results at least till $t = 3000M$.



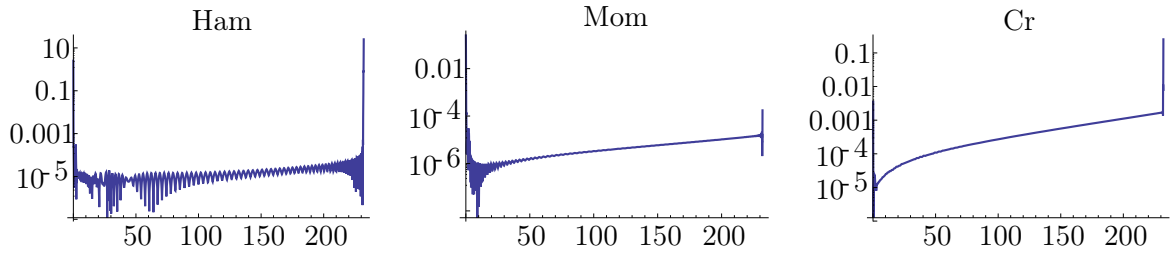


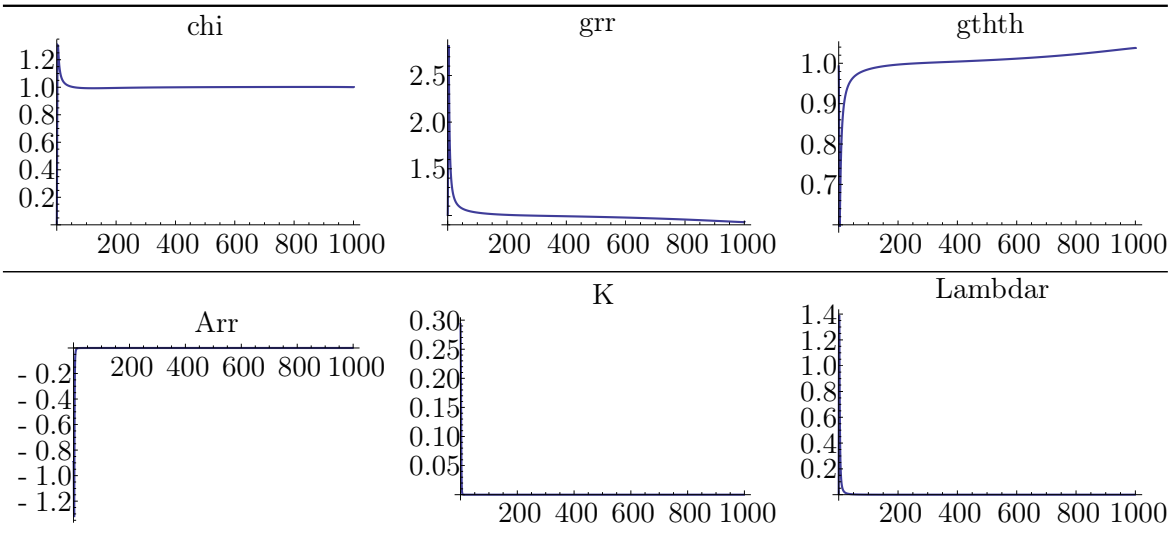
Table 7.2: Interpolated GBSSN variables with wormhole puncture data at $t = 3000M$ with interface radius $r = 2$. The absolute values of the constraints are displayed logarithmically.

We observe that the outer boundary conditions start to interfere the simulation results. The main error contributions are concentrated near to the inner boundary.

7.3 Simulations with significant longer PS grid segments

In order to evaluate the long time behavior of the simulation runs depending on the length of the physical interval, we increase the value of the outer boundary and the PS order while leaving the interface radius unchanged. The grid spacings at the interface take similar values as in the previous run.

Again we consider interface radius $r = 2M$ with 200 FD points. By choosing a PS grid of length 1000 we get similar grid spacings at the interface with PS order 480.



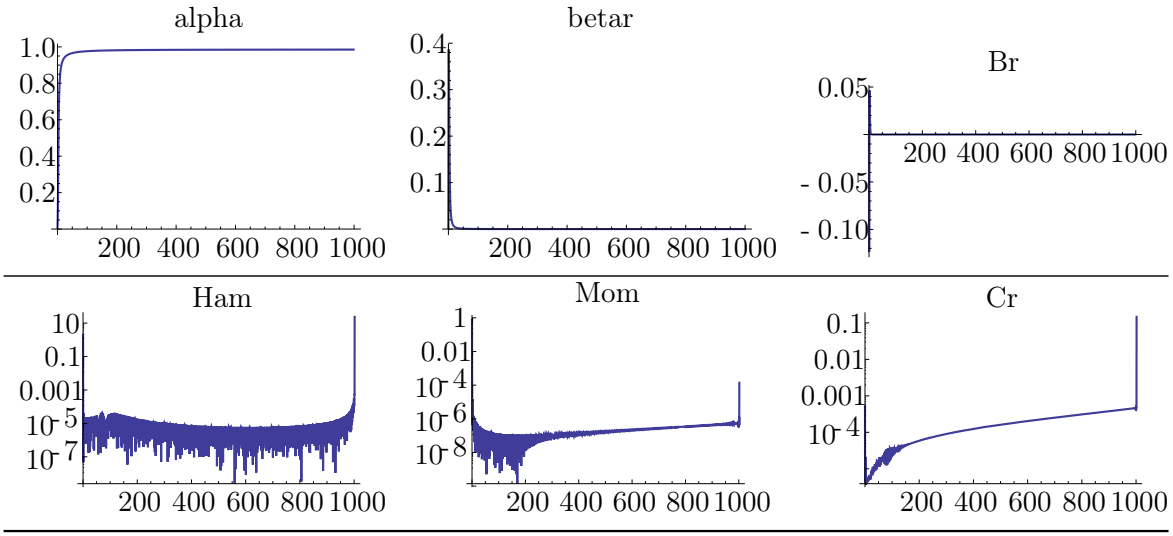


Table 7.3: Interpolated GBSSN variables with wormhole puncture data at $t = 10000M$ with interface radius $r = 2$. The absolute values of the constraints are displayed logarithmically.

We observe that the outer boundary conditions start to fail soon after $t = 10000M$. This can be improved by increasing the length of the FD grid segment, e.g. a simulation run with interface radius $r = 20M$ and the same PS order is still fine at $t = 15000M$.

7.4 Convergence test

After the implementation of the filtering algorithm we verify that the numerical solution still converges.

dt	**ph. dom. FD**	**ph. dom. PS**
0.00125	$[0, 5]$	$[5, 235]$

Table 7.4: Simulation parameters for all resolutions.

The filter parameters in the following table are chosen according to the considerations in the section about Kreiss-Oliger dissipation and Equation 1.73.

FD spacing Δx	**PS orders**	$\Delta x^3 \cdot \epsilon$
$\frac{1}{60}$	384	0.000001
$\frac{1}{80}$	444	0.00000075
$\frac{1}{100}$	480	0.0000006

Table 7.5: Parameter choices for the different resolutions.

The simulations are compared at $t = 10000M$, which in this case corresponds to $8 \cdot 10^6$ time steps. The calculation of the expected convergence factors for these resolutions give for the PS segment cfPS = 3.3 and for the fourth order FD segment cf4 = 3.6. These factors are nearly indistinguishable in the following logarithmic plots.

We start by showing the convergence plot for the metric variable g_{rr}.

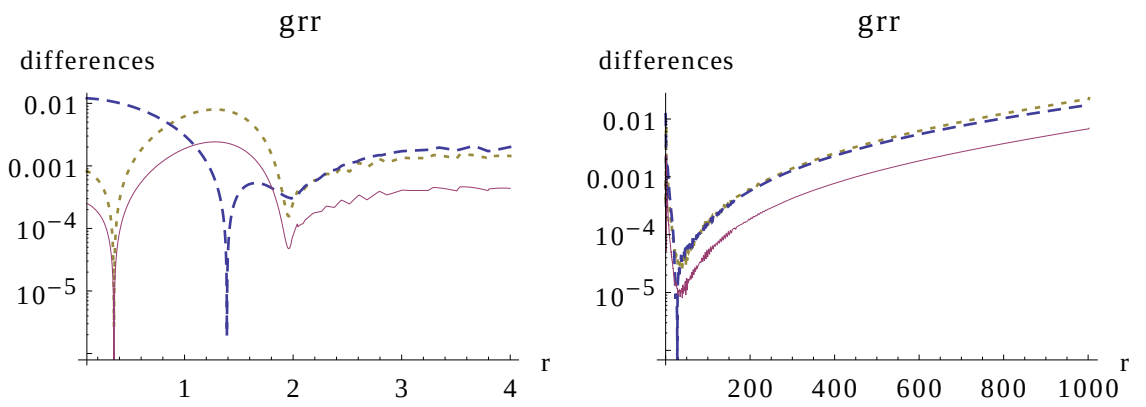


Table 7.6: Metric component g_{rr} at $t = 10000M$, once near to the interface and once on the whole physical interval. The blue dashed and magenta thin lines show the differences between the medium/low and high/medium resolutions. The dotted yellow line is the difference between the higher resolutions multiplied by the convergence factor.

We observe that the errors introduced via the inner boundary condition decrease faster than expected. Near to the interface at $r = 2$ we see a region with larger errors destroying the expected convergence. Most of the PS grid shows convergence with slightly less than the expected PS rate. Additionally we can consider the Constraints.

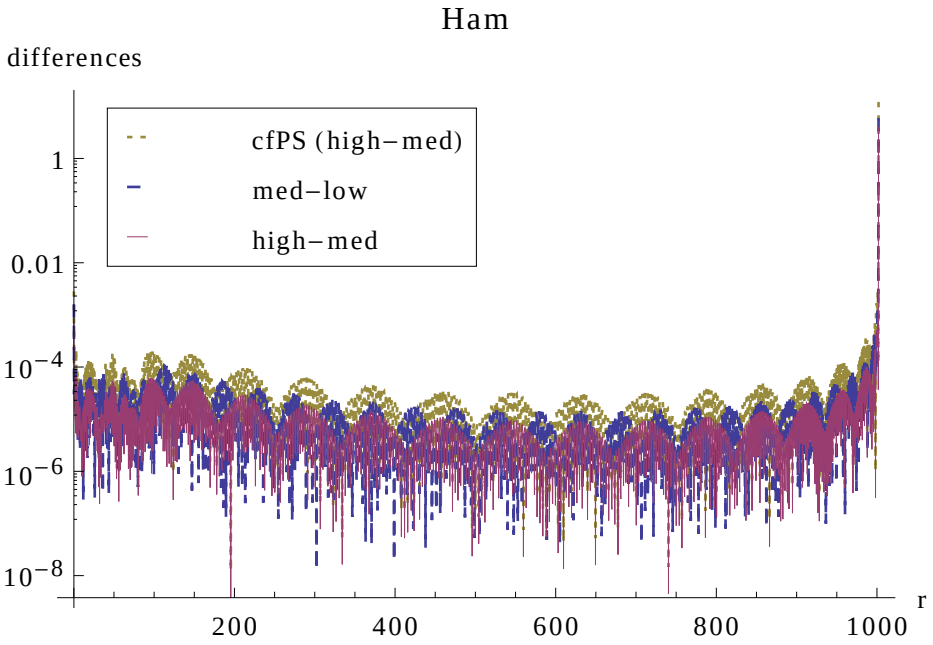


Figure 7.1: Hamiltonian constraint convergence plot at $t = 10000M$

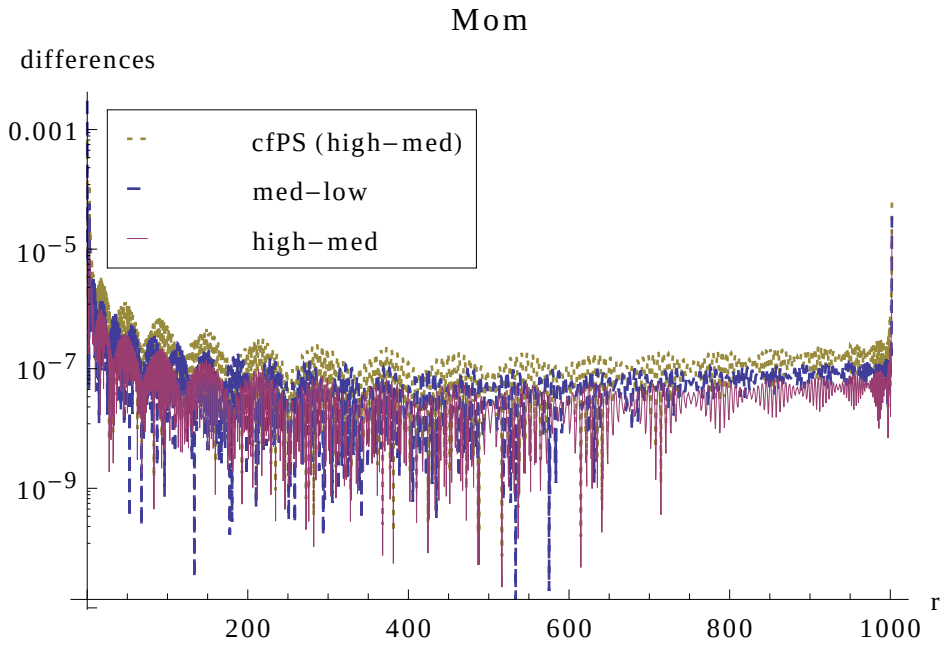


Figure 7.2: Momentum constraint convergence plot at $t = 10000M$

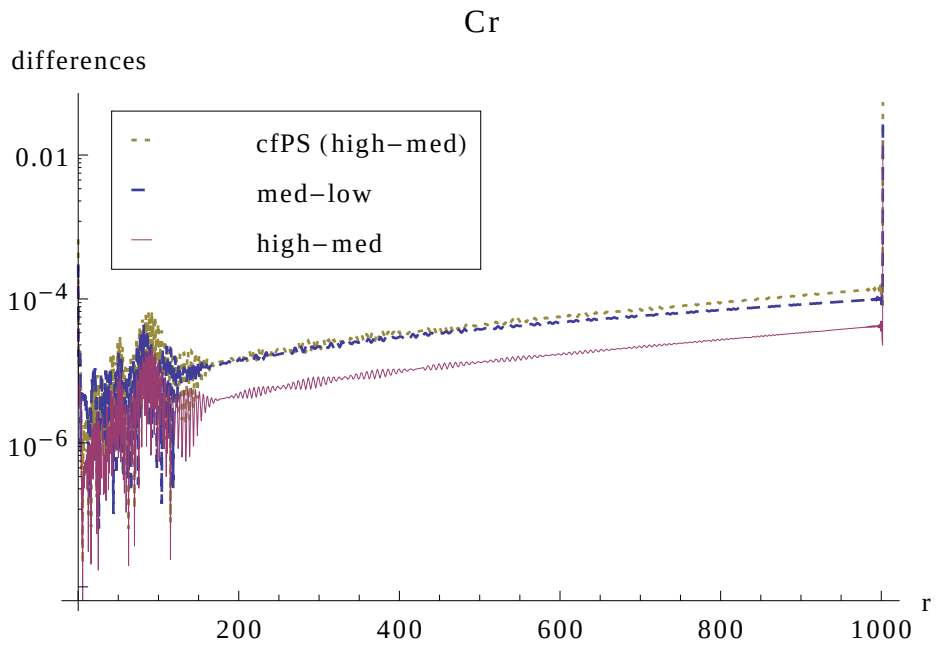


Figure 7.3: C^r constraint convergence plot at $t = 10000M$

We observe that the metric components and the C^r constraint show convergence near to the expected rate. The evaluations for the Hamiltonian and the momentum constraint do not give as nice results. This might be due to the choice of the filter constants.

Similar tests where applied to trumpet puncture initial data. Unfortunately the final time of the simulation decreased to approximately one-third compared to the wormhole puncture case. Also the convergence rates showed worse results compared to the wormhole case. The reason for this behavior is still not clear and a matter of further evaluation.

7.5 Comparison of various long time runs

In order to compare the long time stability of different configurations, we give an overview of various run times. We choose always a PS grid with length 230 and PS order 190. With this configuration we have a slightly higher minimal PS spacing than FD spacing, which is always chosen as 100 grid points per unit interval. According to the choice of the interface radius, also the outer boundary varies. The table shows the final time where the simulation crashed.

We observe definitely longer run times with wormhole puncture initial data than in the case of trumpet data. Also the increasing length of the FD grid segment has positive effects on the stability of the test runs.

Initial data	intf. pos.	$\approx t[M]$
Trumpet	2	500
	5	1300
	10	1680
	15	1720
Wormhole	2	3000
	5	3340
	15	4600
	20	4860

Table 7.7: Comparison of various long time runs

7.6 Conclusion

- Filtering of the FD grid segments leads to an impressive gain of long time stability. This works even without introducing any filtering techniques for the PS grid segment.
- Also increasing the FD interval length improves the long time behavior of the simulations. One reason might be that the filtering takes place only in this region.
- We chose for the Gamma-driver constant $\eta = 2/M$ in case of the wormhole puncture initial data. This lead to better results with the symmetry/antisymmetry conditions at the origin than higher or lower values.
- In general, the wormhole puncture initial data behaves much better than the trumpet initial data. The reason for this result is not clear. For long time test runs with very large PS regions we chose only wormhole puncture initial data.
- The length of the physical interval has significant influence on the run times. The main reasons are the better behavior of radiation boundary conditions farther outside and longer travel times for the errors from one end of the interval to the other.
- Enlarging the physical grid together with a reasonable choice of the FD grid segment length allows run times of more than $t = 15000M$. With interface radius $r = 2$ we always achieved more than 10 light crossing times of the physical interval. With longer FD segments this number increased to 15.

Bibliography

[1] Miguel Alcubierre. Introduction to 3+1 Numerical Relativity. Oxford University Press, 2008.

[2] John G. Baker, Joan Centrella, Dae-Il Choi, Michael Koppitz, and James van Meter. Gravitational-wave extraction from an inspiraling configuration of merging black holes. Phys. Rev. Lett., 96:111102, Mar 2006.

[3] Thomas W. Baumgarte and Stephen G. Naculich. Analytic representation of a black hole puncture solution. Physical Review D, 75(067502), 2007.

[4] Thomas W. Baumgarte and Stuart L. Shapiro. Numerical integration of Einstein's field equations. Phys. Rev. D, 59:024007, Dec 1998.

[5] Thomas W. Baumgarte and Stuart L. Shapiro. Numerical Relativity: Solving Einstein's Equations on the Computer. Cambridge University Press, 2010.

[6] Marsha J. Berger and Joseph Oliger. Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations. Journal of Computational Physics, 53:484–512, 1984.

[7] John P. Boyd. Chebyshev and Fourier Spectral Methods. DOVER Publications, Inc., 2000. Second Edition.

[8] J. David Brown. BSSN in Spherical Symmetry. Classical and Quantum Gravity, 25(20), 2008.

[9] J. David Brown. Covariant formulation of Baumgarte, Shapiro, Shibata and Nakamura and the standard gauge. Physical Review, D 79(104029), 2009.

[10] J. David Brown, Olivier Sarbach, Erik Schnetter, Manuel Tiglio, Peter Diener, Ian Hawke, and Denis Pollney. Excision without excision. Physical Review, D 76(081503), 2007.

[11] Bernd Brügmann. Binary Black Hole Mergers in 3D Numerical Relativity. International Journal of Modern Physics D, 08, 1999.

[12] Bernd Brügmann, Wolfgang Tichy, and Nina Jansen. Numerical Simulation of Orbiting Black Holes. Phys. Rev. Lett., 92:211101, May 2004.

[13] M. Campanelli, C. O. Lousto, P. Marronetti, and Y. Zlochower. Accurate evolutions of orbiting black-hole binaries without excision. Phys. Rev. Lett., 96:111101, Mar 2006.

[14] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang. Spectral Methods: Fundamentals in Single Domains. Springer, 2006.

[15] Lawrence C. Evans. Partial Differential Equations, volume 19 of Graduate Studies in Mathematics. American Mathematical Society, University of California, Berkeley, CA, 2010. Second Edition.

[16] Scott E. Field, Jan S. Hesthaven, Stephen R. Lau, and Abdul H. Mroue. Discontinuous Galerkin method for the spherically reduced BSSN system with second-order operators. Physical Review, D 82, 2010.

[17] Bengt Fornberg. Generation of Finite Difference Formulas on Arbitrary Spaced Grids. Mathematics of Computation, 51(184):699–706, 1988.

[18] Helmut Friedrich. Hyperbolic reductions for Einstein's equations. Classical Quantum Gravity, 13:1451, 1996.

[19] Bertil Gustafsson, Heinz-Otto Kreiss, and Joseph Oliger. Time dependent problems and difference methods, volume 19 of Pure and applied mathematics. John Wiley and sons, Inc., 1995.

[20] Mark Hannam, Sascha Husa, Frank Ohme, Bernd Brügmann, and Niall Ó Murchadha. Wormholes and trumpets: the Schwarzschlild spacetime for the moving-puncture generation. Physical Review D, 78(064020), 2008.

[21] Lawrence E. Kidder, Mark A. Scheel, Saul A. Teukolsky, Eric D. Carlson, and Gregory B. Cook. Black hole evolution by spectral methods. Phys. Rev. D, 62:084032, Sep 2000.

[22] Heinz-Otto Kreiss and Jens Lorenz. Initial-Boundary Value Problems and the Navier-Stokes Equation. Academic Press, Inc., 1989.

[23] Takashi Nakamura, Kenichi Oohara, and Yasufumi Kojima. General Relativistic Collapse to Black Holes and Gravitational Waves from Black Holes. Progress of Theoretical Physics Supplements, 90:1–218, May 1987.

[24] Frans Pretorius. Evolution of binary black-hole spacetimes. Phys. Rev. Lett., 95:121101, Sep 2005.

[25] Frans Pretorius. Numerical relativity using a generalized harmonic decomposition. Classical and Quantum Gravity, 22:425, 2005.

[26] Carl Runge. Über empirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten. Zeitschrift für Mathematik und Physik, 46:224–243, 1901.

[27] Masaru Shibata and Takashi Nakamura. Evolution of three-dimensional gravitational waves: Harmonic slicing case. Phys. Rev. D, 52:5428–5444, Nov 1995.

[28] Béla Szilágyi, Lee Lindblom, and Mark A. Scheel. Simulations of binary black hole mergers using spectral methods. Phys. Rev. D, 80:124010, Dec 2009.

[29] J. W. Thomas. Numerical Partial Differential Equations: Finite Difference Methods. Springer, 1995.

[30] Robert M. Wald. General Relativity. The University of Chicago Press, 1984.

Abstract

The BSSN formalism is a modification of the well known ADM formalism of general relativity. It satisfies the strong hyperbolicity condition and therefore leads to more stable results in numerical problems.

For the simulation of black hole systems using the BSSN formulation of Einstein's equations (wormhole) puncture initial data are usually employed. The simulations are usually implemented using a finite difference approach, although a spectral method may lead to reduced computation costs and enhanced accuracy.

The aim of this thesis is to combine the advantages of both finite difference (FD) and pseudospectral (PS) methods. In order to do that we define a *touching* interface between the FD and PS grid patches and pass information along according to the characteristic fields of the BSSN system.

Rather than attacking the full-blown 3-dimensional problem, we restrict ourselves to spherical symmetry. This leads to a tractable system of equations that can be solved on a workstation. So far, solving puncture evolution without excision or smoothing of the black hole interior has not been achieved with spectral methods.

This thesis implements an accurate and stable hybrid numerical method for the second order in space BSSN system in spherical symmetry. A finite difference grid near the origin is connected by an interface to a pseudospectral grid in the outer part of the physical interval. We use wormhole and trumpet initial data for the simulations. With the help of filtering techniques we are able to stably evolve the BSSN system in spherical symmetry with wormhole puncture initial data for run times longer than $t = 15000M$.

Zusammenfassung

Beim BSSN Formalismus handelt es sich um eine Abwandlung des aus der Literatur bekannten ADM Formalismus der allgemeinen Relativitätstheorie. Im Gegensatz zu ADM erfüllt das BSSN System das Kriterium der starken Hyperbolizität, was in numerischen Simulation zu deutlich stabileren Resultaten führt.

Für die Simulation von Systemen von Schwarzen Löchern mit der BSSN Formulierung der Einstein Gleichungen werden in vielen Fällen Wurmloch-Puncture Anfangsdaten herangezogen. Die Methode der Wahl zur Realisierung der Simulationen ist die der Finiten Differenzen (FD). Nichtsdestotrotz ist zu erwarten, dass spektrale Methoden zu einem geringeren Rechenaufwand und zu erhöhter Genauigkeit beitragen können.

Ziel dieser Arbeit ist eine Kombination der Vorteile von Finiten Differenzen (FD) und pseudospektralen (PS) Methoden. Um dieses zu erreichen, definieren wir ein Interface zwischen FD und PS Gitterregionen, um einen Informationsaustausch mittels charakteristischer Felder des BSSN Systems zu ermöglichen.

Zur einfacheren Handhabbarkeit und Evaluation des Problems beschränken wir uns vorerst auf den Fall der sphärischen Symmetrie. Dies erlaubt die Arbeit mit einem reduzierten Gleichungssystem, welches auch auf einem Workstation-Rechner simuliert werden kann. Bisher konnte das Problem der Simulation von BSSN mit Puncture-Anfangsdaten ohne das Ausschneiden oder Glätten des Inneren des Schwarzen Lochs mit spektralen Methoden allein nicht gelöst werden.

Die in dieser Arbeit vorgestellten Konzepte erlauben eine Implementation einer hybriden Methode für BSSN in sphärischer Symmetrie, um über längere Laufzeiten hinweg stabile Ergebnisse zu erzielen. Nahe dem Zentrum erfolgt die Positionierung eines FD Gittersegments. Daran direkt anschließend und über ein Interface verbunden wird ein PS Gittersegment positioniert, welches den äußeren Teil des physikalischen Intervalls abdeckt. Die Simulationen nutzen sowohl Wurmloch- als auch Trumpet-Puncture Anfangsdaten. Zusätzlich erfolgt die Implementierung von Filtertechniken, um über längere Zeiten stabile Simulationen zu erzielen. Dies führt im Fall von BSSN in sphärischer Symmetrie und Wurmloch-Puncture Anfangsdaten zu Laufzeiten von mehr als $15000M$.

Curriculum Vitae

Contact information

First name:	Kurt
Surname:	Fritz
Address:	Wasagasse 21/2/13, 1090 Wien
Email:	Kurt.Fritz@gmx.at

Personal information

Date of Birth:	24.01.1986
Place of Birth:	Klagenfurt
Citizenship:	Austrian

Education

1992–1996	Elementary School, Pörtschach am Wörthersee
1996–2000	Secondary School, Hauptschule Moosburg
2000–2005	Higher Federal Technical Institute, Höhere Technische Bundeslehranstalt Klagenfurt Mössingerstraße - Technische Informatik und Internet Engineering
25.06.2005	General qualification for university entrance, Reifeprüfung (Matura)
since 10/2006	Study of Physics at the University of Vienna
2007–2012	Study of Mathematics at the University of Vienna

Semester abroad

09/2008 - 01/2009	Student exchange at the Luleå University of Technology, Sweden

Scholarship

2012	Excellence scholarship founded by University of Vienna

Participation

02/2013 3rd Central European Relativity Seminar – Golm

Employment history

07/2002 - 09/2002 Practical course: GREENoneTEC Solarindustrie

07/2003 - 09/2003 Work experience: Strandhotel Prüller - Gastronomie

07/2004 - 09/2004 Practical course: Herbert Fritz - EDV Dienstleistung

07/2005 - 07/2006 Community service: Red Cross

08/2006 - 09/2006 Work experience: Red Cross

10/2009 - 03/2012 Student assistant at the University of Natural Resources and Life Sciences, Vienna