Institut für Physik und Astronomie
Universität Potsdam

und

Max-Planck-Institut für Gravitationsphysik
(Albert-Einstein-Institut) Potsdam

A task-based parallel elliptic solver for numerical relativity with discontinuous Galerkin methods

---

Dissertation

zur Erlangung des akademischen Grades
doctor rerum naturalium (Dr. rer. nat.)

in der Wissenschaftsdisziplin
Gravitationsphysik

vorgelegt der
Mathematisch-Naturwissenschaftlichen Fakultät
der Universität Potsdam

am 1. Juni 2022

von

Nils L. Vu

Tag der Disputation: 26.9.2022

Hauptbetreuer: Prof. Dr. Harald P. Pfeiffer
Zweitbetreuerin: Prof. Dr. Alessandra Buonanno
Mentor: Prof. Dr. Masaru Shibata

Gutachter*innen:

1. Prof. Dr. Harald P. Pfeiffer
Max-Planck-Institut für Gravitationsphysik (Albert-Einstein-Institut) Potsdam
2. Prof. Dr. Helvi Witek
University of Illinois at Urbana-Champaign
3. Prof. Dr. Bernd Brügmann
Friedrich-Schiller-Universität Jena

Published online on the
Publication Server of the University of Potsdam:
https://doi.org/10.25932/publishup-56226
https://nbn-resolving.org/urn:nbn:de:kobv:517-opus4-562265

A task-based parallel elliptic solver for numerical relativity with discontinuous Galerkin methods

Nils L. Vu

Max Planck Institute for Gravitational Physics (Albert Einstein Institute), Am Mühlenberg 1, 14476 Potsdam, Germany

Scientific abstract Elliptic partial differential equations are ubiquitous in physics. In numerical relativity—the study of computational solutions to the Einstein field equations of general relativity—elliptic equations govern the initial data that seed every simulation of merging black holes and neutron stars. In the quest to produce detailed numerical simulations of these most cataclysmic astrophysical events in our Universe, numerical relativists resort to the vast computing power offered by current and future supercomputers. To leverage these computational resources, numerical codes for the time evolution of general-relativistic initial value problems are being developed with a renewed focus on parallelization and computational efficiency. Their capability to solve elliptic problems for accurate initial data must keep pace with the increasing detail of the simulations, but elliptic problems are traditionally hard to parallelize effectively.

In this thesis, I develop new numerical methods to solve elliptic partial differential equations on computing clusters, with a focus on initial data for orbiting black holes and neutron stars. I develop a discontinuous Galerkin scheme for a wide range of elliptic equations, and a stack of task-based parallel algorithms for their iterative solution. The resulting multigrid-Schwarz preconditioned Newton-Krylov elliptic solver proves capable of parallelizing over 200 million degrees of freedom to at least a few thousand cores, and already solves initial data for a black hole binary about ten times faster than the numerical relativity code `SpEC`. I also demonstrate the applicability of the new elliptic solver across physical disciplines, simulating the thermal noise in thin mirror coatings of interferometric gravitational-wave detectors to unprecedented accuracy. The elliptic solver is implemented in the new open-source `SpECTRE` numerical relativity code, and set up to support simulations of astrophysical scenarios for the emerging era of gravitational-wave and multimessenger astronomy.

Zur Lösung von elliptischen Gleichungen der numerischen Relativität auf Supercomputern

Nils L. Vu

Max-Planck-Institut für Gravitationsphysik (Albert-Einstein-Institut),
Am Mühlenberg 1, 14476 Potsdam, Deutschland

Allgemeinverständliche Zusammenfassung Elliptische partielle Differentialgleichungen sind in der Physik allgegenwärtig. Das elektrische Feld einer Ladung, die Gravitation der Erde, die Statik einer Brücke, oder die Temperaturverteilung auf einer heißen Herdplatte folgen trotz verschiedenster zugrundeliegender Physik elliptischen Gleichungen ähnlicher Struktur, denn es sind statische, also zeitunabhängige Effekte. Elliptische Gleichungen beschreiben auch astrophysikalische Szenarien von kataklysmischen Ausmaßen, die jegliche Gegebenheiten auf der Erde weit überschreiten. So werden Schwarze Löcher und Neutronensterne – zwei mögliche Endstadien von massereichen Sternen – ebenfalls von elliptischen Gleichungen beschrieben. In diesem Fall sind es Einstein's Feldgleichungen von Raum, Zeit, Gravitation und Materie. Da Schwarze Löcher und Neutronensterne mehr Masse als unsere Sonne auf die Größe einer Stadt wie Potsdam komprimieren übernimmt die Gravitation, und damit Einstein's allgemeine Relativitätstheorie, die Kontrolle. Es ist die Aufgabe der numerischen Relativität, Szenarien wie die Kollision solcher gewaltigen Objekte mithilfe von Supercomputern zu simulieren und damit die Gravitationswellensignale vorherzusagen, die von Detektoren auf der Erde gemessen werden können. Jede dieser Simulationen beginnt mit Anfangsdaten, die elliptische Gleichungen erfüllen müssen.

In dieser Dissertation entwickle ich neue numerische Methoden um elliptische partielle Differentialgleichungen auf Supercomputern zu lösen, mit besonderem Augenmerk auf Anfangsdaten für Simulationen von Schwarzen Löchern und Neutronensternen. Ich entwickle dafür eine sogenannte *discontinuous Galerkin* Methode um elliptische Gleichungen auf Computern zu repräsentieren, sowie eine Reihe von Algorithmen um diese Gleichungen anschließend schrittweise numerisch zu lösen bis sie die notwendige Präzision erfüllen. Die Besonderheit dieser Algorithmen liegt in ihrer Eigenschaft, in viele Teilprobleme zerlegt auf einer großen Zahl von Rechenkernen parallel arbeiten zu können. Dieses *task-based parallelism* ermöglicht die effektive Verwendung von Supercomputern. Ich demonstriere die Fähigkeit meiner Algorithmen, Berechnungen von über 200 Millionen Unbekannten mit hoher Effizienz auf mindestens einige Tausend Rechenkerne verteilen zu können, und Anfangsdaten zweier sich umkreisender Schwarzer Löcher bereits etwa zehnmal schneller zu lösen als der langjährig verwendete Computercode `SpEC`. Außerdem zeige ich, dass mein neuer Code auch außerhalb der Relativitätstheorie anwendbar ist. Dazu simuliere ich thermisches Rauschen in den Beschichtungen von Spiegeln, das ebenfalls von elliptischen Gleichungen beschrieben wird. Solche Spiegel sind Objekt großen Forschungsinteresses, da sie ein zentrales Element von Gravitationswellendetektoren darstellen. Mein Code zur numerischen Lösung elliptischer Gleichungen ist Teil des kollaborativen und quelloffenen `SpECTRE` Forschungsprojekts zur Simulation astrophysikalischer Szenarien für die aufstrebende Ära der Gravitationswellen- und Multimessenger-Astronomie.

The following anecdote is told of William James. [...] After a lecture on linear and nonlinear solvers, James was accosted by a little old lady.

"Your algorithm that constructs the matrix explicitly, and then solves the linear problem directly, has a very convincing ring to it, Mr. James, but it's inefficient. I've got a better algorithm," said the little old lady.

"And what is that, madam?" inquired James politely.

"That we solve the linear problem iteratively, by means of a preconditioner at every iteration of the algorithm."

Not wishing to point out the absurdity of this little proposition, James decided to gently dissuade his opponent by making her see some of the inadequacies of her position.

"In your algorithm, madam," he asked, "how does this preconditioner find a solution?"

"You're a very clever man, Mr. James, and that's a very good question," replied the little old lady, "but I have an answer to it. And it's this: the preconditioner dispatches to a second, far better, preconditioner, which solves the linear problem in every iteration of the first."

"But how does this second preconditioner find a solution?" persisted James patiently.

To this, the little old lady crowed triumphantly,

"It's no use, Mr. James—it's preconditioners all the way down."

— Adapted from *J. R. Ross, Constraints on Variables in Syntax, 1967*

Contents

Publication list

Lead-author publications:

[1] **N. L. Fischer** and H. P. Pfeiffer, "Unified discontinuous Galerkin scheme for a large class of elliptic equations," Phys. Rev. D **105**, 024034 (2022), arXiv:2108.05826, Chapter 2 of this thesis.

[2] **N. L. Vu**, H. P. Pfeiffer, G. S. Bonilla, N. Deppe, F. Hébert, L. E. Kidder, G. Lovelace, J. Moxon, M. A. Scheel, S. A. Teukolsky, W. Throwe, N. A. Wittek, and T. Włodarczyk, "A scalable elliptic solver with task-based parallelism for the SpECTRE code," Phys. Rev. D **105**, 084027 (2022), arXiv:2111.06767, Chapter 3 of this thesis.

[3] **N. L. Vu**, S. Rodriguez, T. Włodarczyk, G. Lovelace, H. P. Pfeiffer, G. S. Bonilla, N. Deppe, F. Hébert, L. E. Kidder, J. Moxon, and W. Throwe, "High-accuracy numerical models of Brownian thermal noise in thin mirror coatings," Submitted to Phys. Rev. D, Nov. 2021, arXiv:2111.06893, Chapter 5 of this thesis.

Co-author publications:

[4] S. Ma, Q. Wang, N. Deppe, F. Hébert, L. E. Kidder, J. Moxon, W. Throwe, **N. L. Vu**, M. A. Scheel, and Y. Chen, "Gravitational-wave echoes from numerical-relativity waveforms via spacetime construction near merging compact objects," Phys. Rev. D **105**, 104007 (2022), arXiv:2203.03174

[5] L. M. Zertuche, K. Mitman, N. Khera, L. C. Stein, M. Boyle, N. Deppe, F. Hébert, D. A. B. Iozzo, L. E. Kidder, J. Moxon, H. P. Pfeiffer, M. A. Scheel, S. A. Teukolsky, W. Throwe, and **N. L. Vu**, "High precision ringdown modeling: multimode fits and BMS frames," Phys. Rev. D **105**, 104015 (2022), arXiv:2110.15922

[6] J. Moxon, M. A. Scheel, S. A. Teukolsky, N. Deppe, F. Hébert, L. E. Kidder, W. Throwe, and **N. L. Vu**, "The SpECTRE Cauchy-characteristic evolution system for rapid, precise waveform extraction," Submitted to Phys. Rev. D, Oct. 2021, arXiv:2110.08635

[7] N. Deppe, F. Hébert, L. E. Kidder, W. Throwe, I. Anantpurkar, C. Armaza, G. S. Bonilla, M. Boyle, H. Chaudhary, M. D. Duez, F. Foucart, M. Giesler, J. S. Guo, Y. Kim, P. Kumar, I. Legred, D. Li, G. Lovelace, S. Ma, A. Macedo, D. Melchor, M. Morales, J. Moxon, K. C. Nelli, E. O'Shea, H. P. Pfeiffer, T. Ramirez, H. R. Rüter, J. Sanchez, M. A. Scheel, S. Thomas, D. Vieira, **N. L. Vu**, N. A. Wittek, T. Włodarczyk, and S. A. Teukolsky, "Simulating magnetized neutron stars with discontinuous Galerkin methods," Submitted to Phys. Rev. D, Sept. 2021, arXiv:2109.12033

[8] T. Vincent, H. P. Pfeiffer, and **N. L. Fischer**, "hp-adaptive discontinuous Galerkin solver for elliptic equations in numerical relativity," Phys. Rev. D **100**, 084052 (2019), arXiv:1907.01572

[9] M. Boyle et al. (SXS), "The SXS collaboration catalog of binary black hole simulations," Classical and Quantum Gravity **36**, 195006 (2019), arXiv:1904.04831

Software:

[10] N. Deppe, W. Throwe, L. E. Kidder, **N. L. Vu**, F. Hébert, J. Moxon, C. Armaza, G. S. Bonilla, Y. Kim, P. Kumar, G. Lovelace, A. Macedo, K. C. Nelli, E. O'Shea, H. P. Pfeiffer, M. A. Scheel, S. A. Teukolsky, N. A. Wittek, et al., *SpECTRE*, spectre-code.org, version 2022.04.04, Apr. 2022

[11] **N. L. Vu**, *dgpy*, 10.5281/zenodo.5086181, version 0.1, July 2021

[12] **N. L. Vu**, *gwpv*, github:nilsvu/gwpv

Earlier or unrelated publications:

[13] M. Düll, **N. L. Fischer**, B. M. Schaefer, and F. P. Schuller, "Symmetric gravitational closure," 2020, arXiv:2003.07109

[14] V. Kuznetsov, **N. L. Fischer**, and Y. Guo, "The archive solution for distributed workflow management agents of the CMS experiment at LHC," Computing and Software for Big Science **2**, 10.1007/s41781-018-0005-0 (2018), arXiv:1801.03872

1 Introduction

Publications

This thesis is based on my lead-author publications. Chapters 2, 3 and 5 correspond to Refs. [1–3]. This introduction, as well as the discussions in Chapters 4 and 6, place my research in context and present as yet unpublished results.

Numerical simulations of black holes and neutron stars help us understand space, time, gravity, and their interaction with matter. These fundamental concepts are the subject of Einstein's theory of general relativity (GR), which describes the interaction of spacetime and matter as a set of partial differential equations (PDEs). In numerical relativity (NR) we solve these equations on supercomputers. This computational approach allows us to study cataclysmic events in our Universe that are inaccessible to laboratories on earth, such as merging black holes and neutron stars, and predict their outcome.

I find particularly fascinating about the computational approach that it requires the programmer to build up a profound understanding of the physics at play because the computer allows no room for interpretation or for imprecision.

In this thesis I develop new computational methods to solve a subset of the Einstein equations, the elliptic *constraint equations*. As I will outline in this introductory chapter, we require solutions to the Einstein constraint equations to begin every simulation of merging black holes and neutron stars with valid initial data. However, obtaining solutions to the constraint equations is a computational challenge. First, it involves formulating a numerical representation of the elliptic equations on a computational grid, which is the subject of Chapter 2. The numerical representation comes down to a series of matrix equations that computers can, in principle, solve numerically, but that quickly become too large for any ordinary computer to handle. Therefore, advanced computational methods are employed to solve these large systems of equations on computing clusters. Developing such methods for the purpose of generating initial data involving black holes and neutron stars has been the main focus of my work, and is the subject of Chapter 3. Consequently, I might as well have titled this thesis "Inverting matrices on supercomputers". The two main novelties of my work are the development of discontinuous Galerkin (DG) methods for the numerical representation of the Einstein constraint equations [1], and the development of task-based parallel iterative algorithms for their solution [2].

A considerable part of my work has been the implementation of these computational methods to solve elliptic PDEs in the new open-source numerical relativity code `SpECTRE` [10]. I have constructed the initial data solver of the code, which seeds our evolutions of black holes and neutron stars with constraint-satisfying data. I survey the capabilities of the initial data solver, discuss new approaches in the field, and point to ongoing and future work on the subject in Chapter 4.

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

[2]: **Vu** et al. (2022), *A scalable elliptic solver with task-based parallelism for the `SpECTRE` code*. Chapter 3 of this thesis.

[10]: SpECTRE, spectre-code.org

My work also has applications beyond numerical relativity, since elliptic PDEs are common in many areas of physics. One such application is the simulation of thermal noise in thin mirror coatings, as they are employed in interferometric gravitational wave detectors. I explore the applicability of my new elliptic solver to this problem in Chapter 5 and Ref. [3]. I conclude in Chapter 6.

[3]: **Vu** et al. (2021), *High-accuracy numerical models of Brownian thermal noise in thin mirror coatings*. Chapter 5 of this thesis.

1.1 The big picture: numerical relativity in context

At the moment the scientific community is moving into an era of precision gravitational-wave and multi-messenger astronomy, and numerical relativity plays an essential role in it. In this section I outline the current state of the field and its connection to numerical relativity.

1.1.1 Essentials of gravitational wave theory

Gravitational waves are a feature of general relativity, our theory of gravity and spacetime.[1] General relativity relates the geometry of spacetime, encoded in the metric tensor $g_{\mu\nu}$, to the matter content $T_{\mu\nu}$ by

$$\text{the } \textit{Einstein equations} \quad \mathcal{R}_{\mu\nu} - \frac{1}{2} g_{\mu\nu} \mathcal{R} = 8\pi \mathrm{G_N} T_{\mu\nu}, \tag{1.1}$$

$$\text{with the } \textit{Ricci scalar} \quad \mathcal{R} = g^{\mu\nu} \mathcal{R}_{\mu\nu}, \tag{1.2}$$

$$\text{the } \textit{Ricci tensor} \quad \mathcal{R}_{\mu\nu} = \mathcal{R}^{\alpha}{}_{\alpha\mu\nu}, \tag{1.3}$$

$$\text{the } \textit{Riemann tensor} \quad \mathcal{R}^{\alpha}{}_{\beta\mu\nu} = \partial_\mu \Gamma^{\alpha}_{\nu\beta} - \partial_\nu \Gamma^{\alpha}_{\mu\beta} + \Gamma^{\alpha}_{\mu\gamma} \Gamma^{\gamma}_{\nu\beta} - \Gamma^{\alpha}_{\nu\gamma} \Gamma^{\gamma}_{\mu\beta}, \tag{1.4}$$

$$\text{and the } \textit{Christoffel symbols} \quad \Gamma^{\alpha}_{\mu\nu} = \frac{1}{2} g^{\alpha\beta} \left(\partial_\mu g_{\nu\beta} + \partial_\nu g_{\beta\mu} - \partial_\beta g_{\mu\nu} \right). \tag{1.5}$$

1: For introductions to gravitational radiation in general relativity see, e.g., Chapter 7 in Carroll [15] or Chapter 27 in Thorne and Blandford [16].

[15]: Carroll (2004), *Spacetime and Geometry*

[16]: Thorne and Blandford (2017), *Modern Classical Physics*

Here, I have set $c = 1$ but retained the gravitational constant $\mathrm{G_N}$ for illustration. I will also set $\mathrm{G_N} = 1$ in the following. Greek letters represent four-dimensional spacetime indices, latin indices represent three-dimensional spatial indices, and repeated indices are summed over.

Gravitational waves constitute perturbations in the metric tensor,

$$g_{\mu\nu} = \eta_{\mu\nu} + h_{\mu\nu}, \qquad \left| h_{\mu\nu} \right| \ll 1, \tag{1.6}$$

where $\eta_{\mu\nu}$ is the flat Minkowski metric and $h_{\mu\nu}$ represents a small perturbation. By a choice of coordinates we can formulate the metric perturbations in transverse traceless (TT) gauge,

$$h_{0\nu} = 0, \quad \eta^{\mu\nu} h_{\mu\nu} = 0, \quad \partial_\mu h^{\mu\nu} = 0, \tag{1.7}$$

so they are spatial, traceless, and transverse. Under these conditions the Einstein equations (1.1) in vacuum reduce to a wave equation for the metric perturbations,[2]

$$\Box h_{\mu\nu} = 0. \tag{1.8}$$

2: Eq. (7.91) in Carroll [15]

where $\Box = \eta^{\mu\nu}\partial_\mu\partial_n u$ is the flat wave operator. Solutions to these linearized Einstein equations are called *gravitational waves*. They have the form

$$h_{\mu\nu} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & h_+ & h_\times & 0 \\ 0 & h_\times & -h_+ & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{1.9}$$

in TT gauge aligned with the propagation direction, where $h_{+,\times}$ represent the two polarizations of the gravitational wave (Fig. 1.1). The two polarizations are also often combined and expanded in spin-weighted spherical harmonics ${}_{-2}Y_{lm}$,[3]

$$h = h_+ - i h_\times = \frac{1}{r}\sum_{l=2}^{\infty}\sum_{m=-l}^{l} h_{lm}(t)\,{}_{-2}Y_{lm}(\theta,\phi), \tag{1.10}$$

where $h_{lm}(t)$ are the complex *modes* of the gravitational radiation.

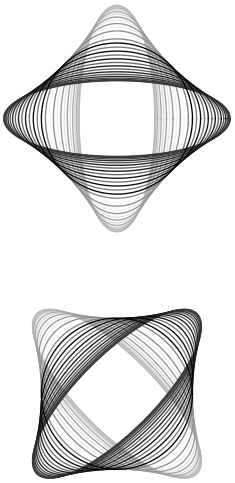


Figure 1.1: Relative motion of a ring of test particles under the effect of a gravitational wave with polarization + (top) and × (bottom). Shown is one oscillation period (black to white).

Gravitational waves are sourced by quadrupolar motion of massive objects. In the presence of matter, the linearized Einstein equations (1.1) reduce to [4]

$$\Box\bar{h}_{\mu\nu} = -16\pi T_{\mu\nu}, \qquad \bar{h}_{\mu\nu} = h_{\mu\nu} - \frac{1}{2}h\eta_{\mu\nu}, \tag{1.11}$$

where $\bar{h}$ is the *trace-reversed* perturbation with $\bar{h} = \eta^{\mu\nu}\bar{h}_{\mu\nu} = -h$, and we have chosen Lorenz gauge $\partial_\mu\bar{h}^{\mu\nu} = 0$. When the source is distant, isolated, and nonrelativistic, the linearized Einstein equations (1.11) have the solution [5]

$$\bar{h}_{ij}(t,\boldsymbol{x}) = \frac{2}{r}\ddot{I}_{ij}(t-r), \tag{1.12}$$

sourced by the accelerated *quadrupole moment* of the matter distribution,

$$I_{ij}(t) = \int x^i x^j T^{00}(t,\boldsymbol{x})\,\mathrm{d}V\,. \tag{1.13}$$

The energy radiated away by gravitational wave emission, or the gravitational wave *luminosity*, is [6]

$$L \equiv -\frac{\mathrm{d}E}{\mathrm{d}t} = \frac{1}{5}\langle\dddot{J}_{ij}\dddot{J}^{ij}\rangle, \tag{1.14}$$

where $J_{ij} = I_{ij} - \frac{1}{3}\delta_{ij}\delta^{kl}I_{kl}$ is the trace-reduced quadrupole moment and the angle brackets denote an average over several wavelengths.

3: See, e.g., Eq. (17.36) and Appendix D in Baumgarte and Shapiro [17].

[17]: Baumgarte and Shapiro (2010), *Numerical Relativity: Solving Einsteins Equations on the Computer*

4: Eq. (7.125) in Carroll [15]

5: Eq. (7.140) in Carroll [15]

6: Eq. (7.189) in Carroll [15]

1.1.2 Gravitational radiation from orbiting binaries

Binary systems of orbiting massive objects produce quadrupole moments, which is the reason why binary black holes and neutron stars are of primary interest in numerical relativity. For instance, consider two objects with masses $M_1 \geq M_2$ in a circular orbit with separation D. We can

define

$$\text{the } \textit{total mass} \quad M = M_1 + M_2, \tag{1.15}$$

$$\text{the } \textit{mass ratio} \quad q = \frac{M_1}{M_2} \geq 1 \tag{1.16}$$

$$\text{and the } \textit{reduced mass} \quad \mu = \frac{M_1 M_2}{M}. \tag{1.17}$$

Assuming a Newtonian orbit, the binary has the Keplarian orbital angular velocity

$$\Omega = \sqrt{\frac{M}{D^3}} \tag{1.18}$$

and the quadrupole moment is [7]

$$I_{xx} = \mu D^2 \cos^2 \Omega t, \qquad I_{yy} = \mu D^2 \sin^2 \Omega t, \tag{1.19a}$$

$$I_{xy} = I_{yx} = \mu D^2 \cos \Omega t \sin \Omega t, \qquad I_{zi} = I_{iz} = 0, \tag{1.19b}$$

assuming the orbit is in the xy plane. This matter distribution generates the metric perturbation

$$\bar{h}_{ij}(t, \boldsymbol{x}) = \frac{4}{r} \mu \frac{M}{D} \begin{pmatrix} -\cos 2\Omega(t-r) & -\sin 2\Omega(t-r) & 0 \\ -\sin 2\Omega(t-r) & \cos 2\Omega(t-r) & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{1.20}$$

by the quadrupole formula (1.12). Evidently, binary systems emit gravitational waves at twice their orbital frequency, with an amplitude proportional to M/D. Since orbiting binaries of objects with radius R can only reach separations $D \gtrsim 2R$ before colliding, the most promising astrophysical source of strong gravitational radiation are binaries of objects with high compactness M/R. Such objects are black holes and neutron stars, with compactnesses of $M/R \sim 0.1$ to 1, making them primary targets of study in numerical relativity.

Due to the emission of gravitational radiation the orbiting binary will *inspiral*, decreasing in separation (Fig. 1.2). For the Newtonian binary, Eq. (1.19), the gravitational wave luminosity (1.14) is [8]

$$L = \frac{32}{5} \frac{G_N^4}{c^5} \left(\frac{\mu}{M}\right)^2 \left(\frac{M}{D}\right)^5, \tag{1.21}$$

where I have restored powers of G_N and c to facilitate magnitude estimates. Approximating the binary as bound by Newtonian gravity, $E = -\mu M/2D$, and assuming it loses energy adiabatically by gravitational radiation, the separation between the two objects will decrease as

$$\dot{D} = -\frac{64}{5} \frac{G_N^3}{c^5} \frac{\mu M^2}{D^3}. \tag{1.22}$$

For example, the separation between the Earth and the Moon decreases by only $\sim 2.3\,\text{fm/kyr}$ due to gravitational radiation.[9] However, two orbiting black holes with total mass $M = M_1 + M_2 = 60\,M_\odot$ at separation $D = 10M$ ($\sim 890\,\text{km}$) approach each other with $\sim 2\,\text{km/ms}$ by Eq. (1.22). During this inspiral the orbital angular frequency rises and the binary will emit ever stronger gravitational radiation, accelerating its demise.

7: Eq. (27.70d) in Thorne and Blandford [16]

(1.12): $\bar{h}_{ij}(t, \boldsymbol{x}) = \frac{2}{r} \ddot{I}_{ij}(t-r)$

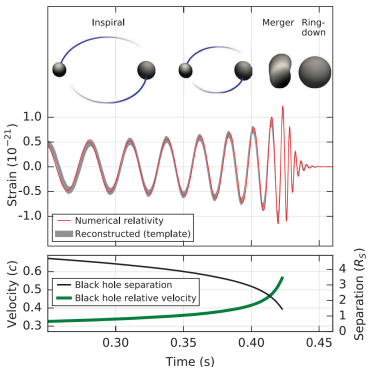


Figure 1.2: Inspiral, merger, and ringdown of a black hole binary (Fig. 2 in Ref. [18]).

[18]: LIGO, Virgo (2016), *Observation of Gravitational Waves from a Binary Black Hole Merger*

8: Eq. (27.73) in Thorne and Blandford [16]

9: The separation between the Earth and the Moon actually *increases* by $\sim 3.8\,\text{cm/yr}$ due to the much more dominant tidal dissipation.

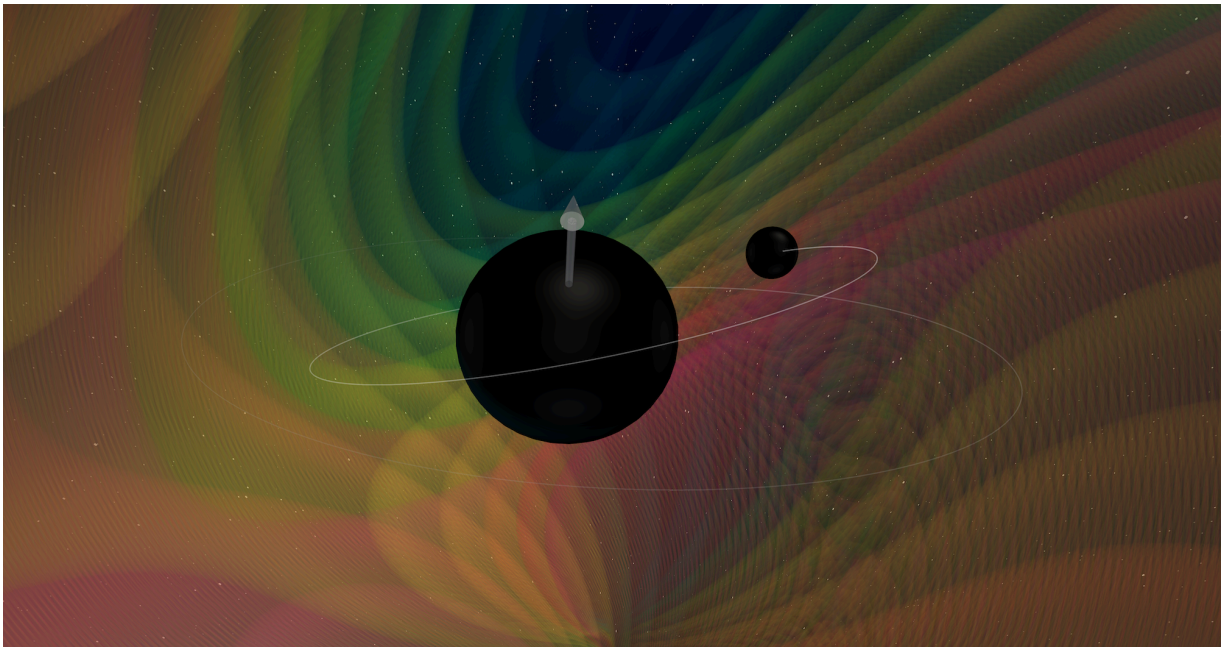


Figure 1.3: Numerical simulation of two black holes that inspiral and merge, emitting gravitational waves. One black hole is 3.5× more massive than the other and spins, which makes the orbit precess. The simulated gravitational wave signal is consistent with the observation made by the LIGO and Virgo gravitational wave detectors on Apr 12, 2019 (`GW190412`). See Appendix A for details on this visualization.

The quadrupole formula (1.12) used to derive Eqs. (1.21) and (1.22) is only the lowest-order ("Newtonian") term of a perturbative series in the velocity of the binary objects. It loses validity as the separation of the binary decreases and the objects gain velocity, entering the relativistic regime. Higher-order terms in the perturbative series are the subject of post-Newtonian (PN) theory [19] and allow to calculate the inspiral of the binary at separations so close that the Newtonian approximation is no longer valid (see also Section 1.1.4).

(1.12): $\bar{h}_{ij}(t, \boldsymbol{x}) = \frac{2}{r}\ddot{I}_{ij}(t - r)$

[19]: Blanchet (2014), *Gravitational Radiation from Post-Newtonian Sources and Inspiralling Compact Binaries*

Inevitably, the binary will enter a strongly relativistic regime at close separations and high velocities, and eventually *merge*. While the inspiral stage of the binary evolution may be described by perturbation theory, the strongly relativistic merger regime is only accessible by numerical simulations that solve the full Einstein equations, often in all four dimensions (Fig. 1.3). This is the arena of **numerical relativity** (NR). Since the first landmark numerical simulations of binary black hole (BBH) mergers in 2005 [20–22], numerical relativity has graduated to an indispensible tool of contemporary gravitational-wave research.[10] The predictive power of BBH, binary neutron star (BNS), and black hole–neutron star (BHNS) simulations makes numerical relativity essential to gravitational-wave astrophysics and the emerging field of multimessenger astronomy, as I will detail in the following sections. Furthermore, numerical relativity provides insights into strongly-relativistic scenarios that are inaccessible by any other method, such as the dynamics of event horizons, light, and matter in the vicinity of the most violent astrophysical events in our Universe. As a flourishing discipline at the intersection of gravitational physics and high-performance computing, many advances in numerical relativity also have interdisciplinary applications across the computational physics community.[11]

[20]: Pretorius (2005), *Evolution of binary black hole spacetimes*

[21]: Campanelli et al. (2006), *Accurate evolutions of orbiting black-hole binaries without excision*

[22]: Baker et al. (2006), *Gravitational wave extraction from an inspiraling configuration of merging black holes*

10: See, e.g., Baumgarte and Shapiro [17] for an introduction to numerical relativity.

[17]: Baumgarte and Shapiro (2010), *Numerical Relativity: Solving Einsteins Equations on the Computer*

11: See Chapter 5 for an interdisciplinary application of the work in this thesis.

In the final stage of binary evolution, the massive remnant of the merger will *ring down* to equilibrium. Binary black holes will merge to a spinning black hole remnant that emits a spectrum of exponentially decaying quasinormal modes (QNMs). When binary neutron stars merge, they either form a stable neutron star remnant or collapse to a black hole, ringing down in the process as well. The gravitational radiation in this ringdown stage can, again, be described perturbatively [23–25]. Instead of perturbing the flat Minkowski metric, as in Eq. (1.6), we perturb the Kerr metric $g_{\mu\nu}^{\text{Kerr}}$ that represents a stationary spinning black hole,[12]

$$g_{\mu\nu} = g_{\mu\nu}^{\text{Kerr}} + h_{\mu\nu}, \qquad \left|h_{\mu\nu}\right| \ll 1, \tag{1.23}$$

where $h_{\mu\nu}$, again, represents the metric perturbations. Following the decomposition (1.10) of $h_{\mu\nu}$ in spin-weighted spherical harmonics, the ringdown signal is a superposition of damped sinusoids,

$$h_{lm}(t) = \sum_{n=0}^{\infty} A_{lmn} e^{-i\sigma_{lmn} t}, \tag{1.24}$$

where the (complex) frequencies σ_{lmn} are known functions of the mass and spin of the remnant black hole, but the (also complex) excitation coefficients A_{lmn} depend on the preceding merger dynamics and hence are determined numerically. Note that QNMs are more naturally expressed in terms of spin-weighted *spheroidal* harmonics related to the Kerr background, rather than the spin-weighted *spherical* harmonics in Eq. (1.10), so the parameters σ_{lmn} and A_{lmn} are related to the Kerr QNM frequencies and excitation coefficients by a transformation between the two functional bases [26].

[23]: Teukolsky (1973), *Perturbations of a rotating black hole. 1. Fundamental equations for gravitational electromagnetic and neutrino field perturbations*

[24]: Buonanno, Cook, and Pretorius (2007), *Inspiral, merger and ring-down of equal-mass black-hole binaries*

[25]: Berti, Cardoso, and Starinets (2009), *Quasinormal modes of black holes and black branes*

12: For a discussion of the Kerr metric see, e.g., Section 6.6 in Carroll [15]. We will encounter the Kerr metric numerous times in this thesis, e.g., in Section 4.1.

(1.10): $h = h_+ - ih_\times = \frac{1}{r}\sum_{l=2}^{\infty}\sum_{m=-l}^{l} h_{lm}(t)\,{}_{-2}Y_{lm}(\theta,\phi)$

[26]: Berti and Klein (2014), *Mixing of spherical and spheroidal modes in perturbed Kerr black holes*

1.1.3 Gravitational-wave astronomy

The first detection of a gravitational wave (GW) signal by the LIGO and Virgo collaborations on Sep 14, 2015 moved gravitational-wave physics from theory to a precision experimental science. Decades of research on gravitational wave detector technology [27] and data analysis methods [28] culminated in this observation, named `GW150914` [18]. Two large laser interferometers with kilometer-long arms located in the northwest and southeast of the USA (Fig. 1.4) picked up a signal from the cataclysmic merger of two black holes. The gravitational radiation emitted by the merger traveled toward Earth at the speed of light for 1.3 billion years. At the sites of the two LIGO instruments, some 3000 km or 10 ms light travel time apart, the gravitational radiation slightly compressed and elongated the $L = 4\,\text{km}$ long laser cavities of the interferometers by a length difference $\delta L(t)$, producing a measurable signal,

$$h(t) \propto \frac{\delta L(t)}{L}, \tag{1.25}$$

the *gravitational wave strain*. Although two merging stellar-mass black holes such as `GW150914` can emit the energy equivalent of multiple solar masses in only a fraction of a second, which is more energy than emitted by all stars in the observable universe combined during this short time, the expected measurable effect is only $h \lesssim 10^{-21}$ for typical signals. The effect is so small because it is diminished by both the magnitude of the

[27]: LIGO (2015), *Advanced LIGO*

[28]: LIGO, Virgo (2020), *A guide to LIGO–Virgo detector noise and extraction of transient gravitational-wave signals*

[18]: LIGO, Virgo (2016), *Observation of Gravitational Waves from a Binary Black Hole Merger*

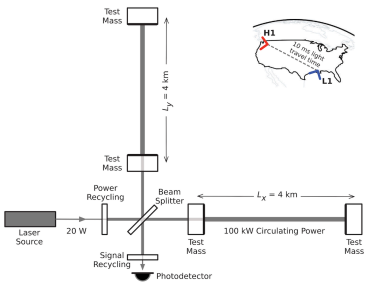


Figure 1.4: Simplified diagram of Advanced LIGO (Fig. 3 in Ref. [18]).

gravitational constant G_N in the Einstein equations, Eq. (1.1), and by the $1/r$ decay in amplitude of the signal traveling through vast amount of space (hundreds of megaparsec for `GW150914`), Eq. (1.12).

Since the measurable gravitational wave strain is so small, it is easily concealed by a plethora of noise effects that disturb the laser interferometers [27, 29] (Fig. 1.5). At low frequencies, below $\sim 20\,\text{Hz}$, the dominant source of noise is *seismic*, caused by vibrations in the ground. Examples include seismic activity, but also storms, lightning strikes, or cars traversing the nearby parking lot (it also does not help that LIGO Livingston in the southeast US is located in a working forest with regularly falling trees). Instrumentalists go to great lengths to isolate the mirrors of the interferometer from seismic effects, employing passive technology such as sophisticated suspension systems, and active technology such as feedback control loops to counteract unwanted motion. At high frequencies, above $\sim 1\,\text{kHz}$, the sensitivity is limited by *quantum shot noise*, caused by the statistical uncertainty in measuring photon arrival rates. Higher laser power in the interferometers can combat this effect, but must be balanced with noise the laser power induces in the mirrors. Specifically, the noise in the most sensitive regime of the detector, around $\sim 100\,\text{Hz}$, is dominated by *Brownian thermal noise* induced by the laser in the thin reflective coatings of the mirrors. Therefore, some detector designs involve cryogenic cooling of the mirrors [30], and materials are being developed that minimize coating thermal noise.[13]

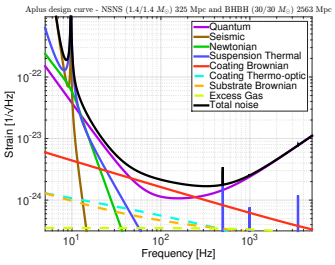


Figure 1.5: Noise sources limiting the sensitivity of the Advanced LIGO detectors (Fig. 2 in Ref. [29]).

[29]: Barsotti et al. (2018), *The A+ design curve*

[30]: KAGRA (2020), *The status of KAGRA underground cryogenic gravitational wave telescope*

To extract the gravitational wave signal from the detector noise, sophisticated data analysis techniques are employed [28]. First, a *search* scans the data stream of each detector for potential signals. The data stream is

$$d(t) = h(t) + n(t), \tag{1.26}$$

where $h(t)$ is the detector response to a gravitational wave signal (Fig. 1.6). The signal is concealed by noise $n(t)$ that is typically much larger in amplitude. However, if we can predict possible gravitational wave signals $h_{ij}(t)$, or *waveforms*, then the *matched filtering* technique allows to find similar signals in the data. The prediction of gravitational waveforms from astrophysical sources is outlined in Section 1.1.4 below. It is an essential contemporary discipline by itself and is supported by numerical relativity in many aspects. From a predicted waveform we can compute the detector response

$$h(t) = F_+(\theta, \phi, \psi)\, h_+(t) + F_\times(\theta, \phi, \psi)\, h_\times(t), \tag{1.27}$$

where $F_{+,\times}$ are the known *antenna patterns* of the detector that depend on the source location in the sky and on the wave polarization ψ. Then, the *signal-to-noise ratio* of the predicted waveform is

$$\text{SNR} = \frac{\big(d(t)\big|h(t)\big)}{\sqrt{\big(h(t)\big|h(t)\big)}}, \tag{1.28}$$

where the noise-weighted inner product is defined as

$$\big(p(t)\big|q(t)\big) := 4\,\text{Re}\int_0^\infty \mathrm{d}f\, \frac{\tilde{p}^*(f)\tilde{q}(f)}{S_n(f)}. \tag{1.29}$$

13: Although seemingly distant to numerical relativity and the topic of this thesis, I approach precisely this problem of Brownian thermal noise in thin mirror coatings from a computational perspective in Chapter 5, since it involves elliptic equations not too different to those found in numerical relativity.

[28]: LIGO, Virgo (2020), *A guide to LIGO–Virgo detector noise and extraction of transient gravitational-wave signals*

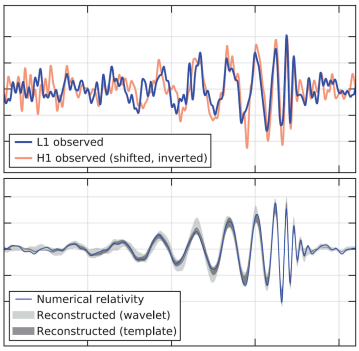


Figure 1.6: Signal and waveform templates for the `GW150914` detection (Fig. 1 in Ref. [18]).

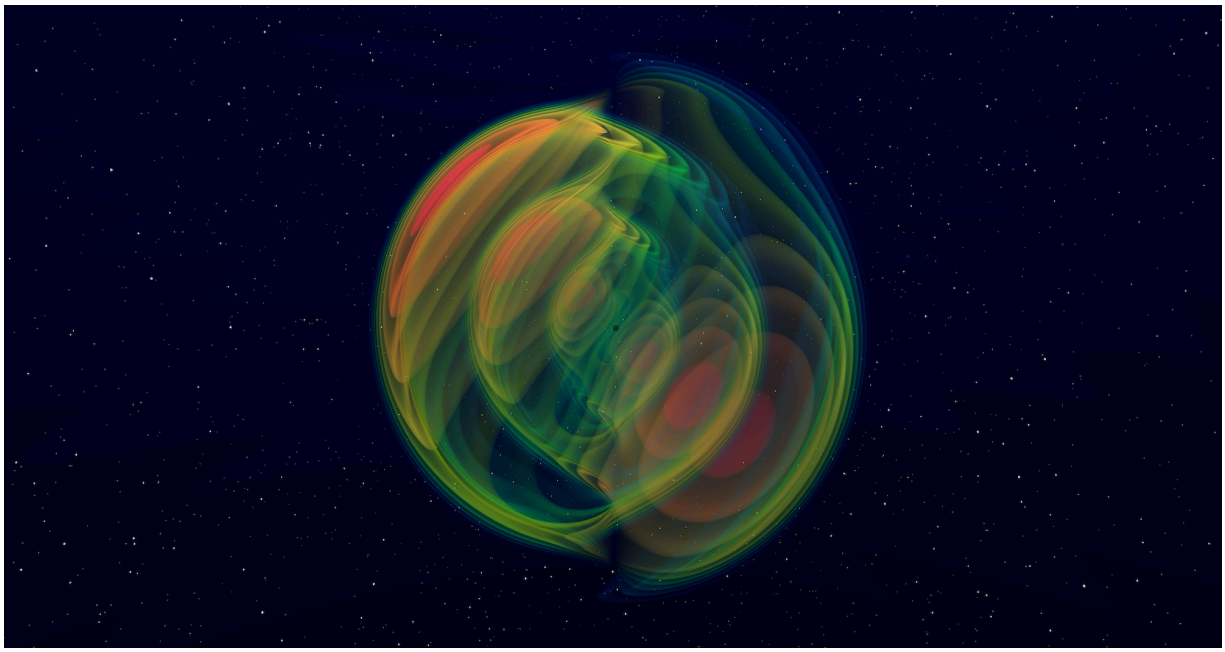


Figure 1.7: Visualization of the `GW190412` event, the first observation of two merging black holes with significantly different masses and an indication for precession (see Appendix A).

Here, $\tilde{p}(f)$ is the Fourier transform of the time series $p(t)$, $\tilde{p}^*$ is its complex conjugate, and $S_n(f)$ is the power spectral density (PSD)

$$S_n(f) = 2\int_{-\infty}^{\infty} \mathrm{d}\tau \, \langle n(t)n(t+\tau)\rangle_t \, e^{-2\pi f\tau}, \tag{1.30}$$

which is the Fourier transform of the autocorrelation function of the noise. It can be estimated from a segment of data with no signal. Note that the SNR is highest when the data stream equals the predicted waveform, $d(t) = h(t)$. In Eq. (1.28) the data stream is *filtered* by $W_h(f) = \tilde{h}(f)/S_n(f)$, the *Wiener filter* or noise-weighted *template*, and the resulting SNR represents a measure of how well the data matches the template. Therefore, in searches for gravitational wave detections we continuously match the data stream with a range of predicted waveforms, a *template bank*, and flag a segment of data as potentially containing a signal when the SNR of a template exceeds a threshold. This matched-filtering technique is complemented by *unmodeled* searches based on wavelet theory, that scan the data stream for excess power without employing predicted waveforms. Furthermore, correlation of signals in multiple detectors are essential to build confidence in detections.

[28]: LIGO, Virgo (2020), *A guide to LIGO–Virgo detector noise and extraction of transient gravitational-wave signals*

Once a potential signal in the data stream has been identified, a *parameter estimation* pipeline analyses the signal in detail and determines the characteristics of its source [28]. This strategy, too, relies on predictions of gravitational waveforms from possible astrophysical sources. Within a Bayesian framework, we estimate the posterior probability density function $p(\boldsymbol{\theta}|d, h)$ that the detector output $d(t)$ is described by a waveform model $h(t;\boldsymbol{\theta})$ with parameters $\boldsymbol{\theta}$. The parameters typically include the masses and spins of the two coalescing objects, the inclination of the orbital plane relative to Earth, the distance to the source, the sky localization, and a phase shift relative to the peak of the signal. By Bayes

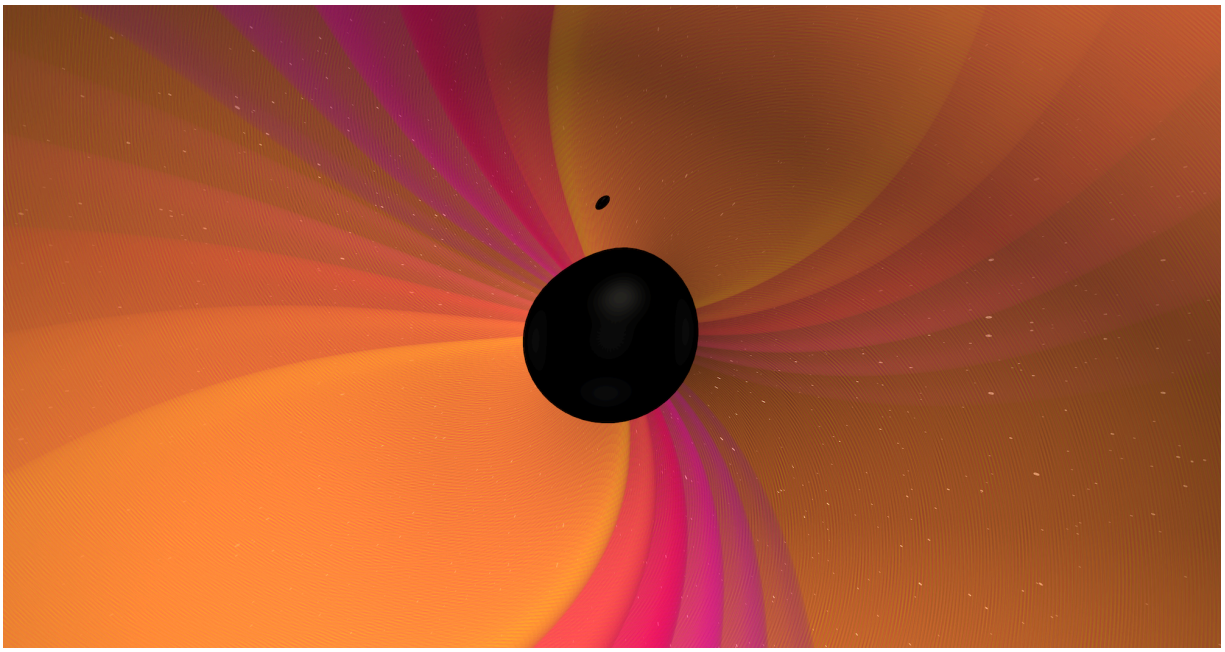


Figure 1.8: Visualization of the `GW190814` event, where the smaller object was either a very light black hole or a very heavy neutron star (see Appendix A).

theorem, the posterior probability for the parameters $\boldsymbol{\theta}$ is

$$p(\boldsymbol{\theta}|d) = \frac{p(d|\boldsymbol{\theta})\,p(\boldsymbol{\theta})}{p(d)}, \tag{1.31}$$

where the *likelihood* is

$$p(d|\boldsymbol{\theta}) = \exp\left(-\frac{1}{2}\sum_{I}^{\text{detectors}}\left[\left(d_I - h_I(\boldsymbol{\theta})\middle|d_I - h_I(\boldsymbol{\theta})\right) + \int \ln S_n^I(f)\,\mathrm{d}f\right]\right). \tag{1.32}$$

It uses the noise-weighted inner product defined by Eq. (1.29) and a sum over all detectors in the network. The *prior* $p(\boldsymbol{\theta})$ encodes assumptions about the parameters, and the *evidence* is the normalization $p(d) = \int \mathrm{d}\boldsymbol{\theta}\, p(\boldsymbol{\theta})p(d|\boldsymbol{\theta})$. To sample the posterior probability, Eq. (1.31), over the high-dimensional space spanned by the parameters $\boldsymbol{\theta}$, a range of computational sampling methods are employed, such as Markov chain Monte Carlo (MCMC) methods. They typically traverse the parameter space stochastically, gravitating toward regions of the parameter space with large posterior probability. Then, quoted parameters for detections like `GW150914` [31] represent marginalizations over the posterior probability distribution to obtain a single number and associated credible intervals for each parameter.

[31]: LIGO, Virgo (2016), *Properties of the Binary Black Hole Merger GW150914*

To date, we have observed a total of 90 gravitational wave events [32–34]. Of these,

- two are confident binary neutron star (BNS) mergers (`GW170817` and `GW190425`),
- three are likely black hole–neutron star (BHNS) mergers (`GW190917_114630`, `GW191219_163120`, and `GW200115_042309`),
- two more are possible BHNS mergers (`GW190814` and `GW200210_092254`),

[32]: LIGO, Virgo (2019), *GWTC-1: A Gravitational-Wave Transient Catalog of Compact Binary Mergers Observed by LIGO and Virgo during the First and Second Observing Runs*

[33]: LIGO, Virgo (2021), *GWTC-2: Compact Binary Coalescences Observed by LIGO and Virgo During the First Half of the Third Observing Run*

[34]: LIGO, Virgo, KAGRA (2021), *GWTC-3: Compact Binary Coalescences Observed by LIGO and Virgo During the Second Part of the Third Observing Run*

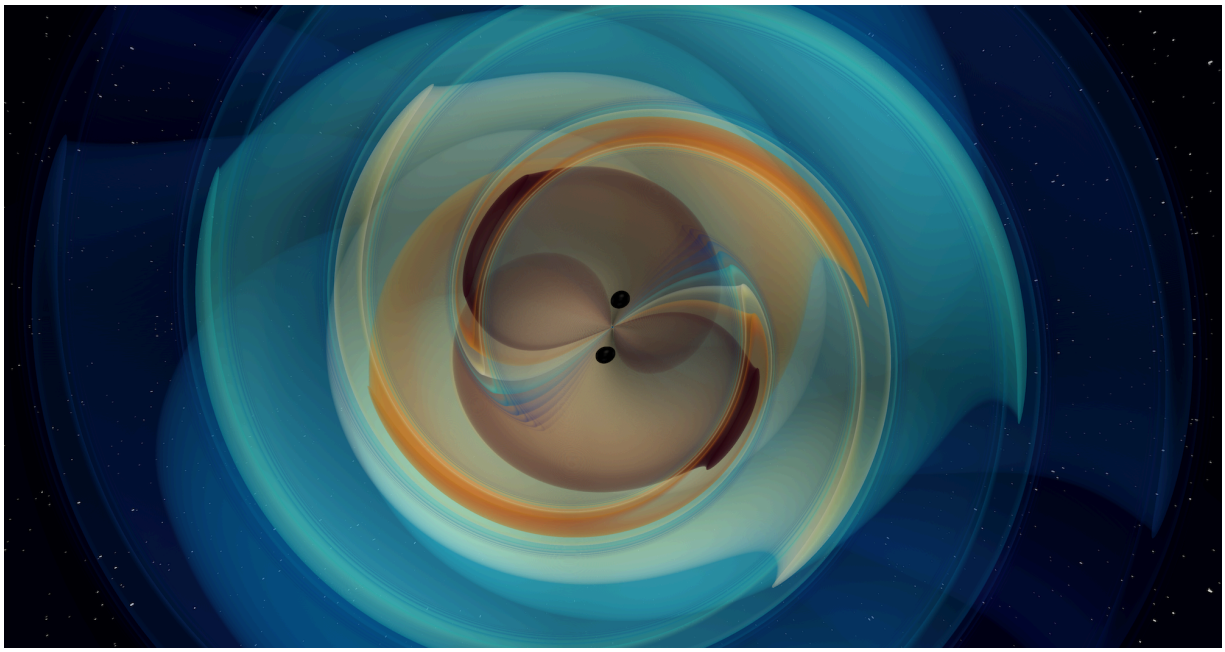


Figure 1.9: Visualization of the `GW190521` event, where two extraordinarily massive black holes merged (see Appendix A).

- and the remaining 83 are confident BBH mergers.

I have produced numerical-relativity visualizations for three exceptional gravitational-wave events (`GW190412` [35], `GW190814` [36], and `GW190521` [37]) to support outreach efforts of the LIGO and Virgo collaborations. Figures 1.3 and 1.7 to 1.9 show snapshots from these visualizations, and are detailed in Appendix A.

[35]: LIGO, Virgo (2020), *GW190412: Observation of a Binary-Black-Hole Coalescence with Asymmetric Masses*
[36]: LIGO, Virgo (2020), *GW190814: Gravitational Waves from the Coalescence of a 23 Solar Mass Black Hole with a 2.6 Solar Mass Compact Object*
[37]: LIGO, Virgo (2020), *GW190521: A Binary Black Hole Merger with a Total Mass of* $150M_\odot$

1.1.4 Waveform models

Gravitational-wave detections and parameter estimation methods rely largely on predictions of gravitational waveforms. However, no single technique discussed in Section 1.1.2 can predict the dynamics and gravitational radiation of a general-relativistic binary in its entirety. Post-Newtonian (PN) approximations are only sufficiently precise during the early inspiral of the binary. Numerical relativity (NR) simulations can cover the late inspiral and the merger, but are too computationally expensive to simulate the large number of orbits that low-mass binaries can spend in the GW detectors' sensitivity bands. It is of course quite unfeasible to perform a full NR simulation for each evaluation of the template in Eq. (1.28) or the likelihood in Eq. (1.31). Therefore, the GW data analysis pipelines employ approximate *waveform models* that are fast to evaluate [28]. These waveform models all rely on data from NR simulations to provide the "ground truth" for their approximations.

[28]: LIGO, Virgo (2020), *A guide to LIGO–Virgo detector noise and extraction of transient gravitational-wave signals*

A large class of waveform models used routinely in GW data analysis is based on semi-analytic approaches, such as the effective-one-body (EOB) framework [38]. These models typically solve the general-relativistic dynamics of an orbiting binary perturbatively, employing PN, post-Minkowskian (PM), and gravitational self-force methods. The perturbative expansions involve parameters that are informed by, and calibrated to, numerical simulations. For example, the `EOBNR` [39, 40] family of

[38]: Buonanno and Damour (1999), *Effective one-body approach to general relativistic two-body dynamics*
[39]: Bohé et al. (2017), *Improved effective-one-body model of spinning, nonprecessing binary black holes for the era of gravitational-wave astrophysics with advanced detectors*
[40]: Ossokine et al. (2020), *Multipolar Effective-One-Body Waveforms for Precessing Binary Black Holes: Construction and Validation*

waveform models, which are routinely used in LIGO data analysis [28], perturbs the dynamics of a test particle in the vicinity of a spinning black hole in the symmetric mass ratio $\nu = \mu/M$, where μ is the reduced mass given by Eq. (1.17). The EOB framework employs a Hamiltonian formulation for the motion of the particle, so the orbital dynamics can be computed numerically using integration methods for ordinary differential equations (ODEs). The EOB Hamiltonian is constructed from resummations of PN, PM [41], and gravitational self-force [42] expansions. NR-calibrated parameters in the EOB model encode effects such as truncating the perturbative expansions at a finite order [39, 43, 44]. Furthermore, the merger-ringdown part of the waveform [39, 45] and higher-order modes in the gravitational radiation [46] are constructed from NR-calibrated parameters.

Other classes of waveform models are more directly based on numerical simulations, such as the `Phenom` [47] and `NRSurrogate` [48–50] families of models. The `Phenom`-type models construct frequency-domain fits of phenomenological parameters to NR waveforms, whereas the `NRSurrogate`-type models smoothly interpolate directly between NR waveforms assuming no underlying phenomenology. Both types of models use semi-analytic PN or EOB approaches to extend their waveforms toward the early inspiral.

Many authors of waveform models emphasize their reliance on accurate NR simulations in their articles [39, 46, 47, 51]. Waveform models are calibrated, their faithfulness verified, and their domain of validity restricted to the set of available NR simulations at the time of their development. For this purpose, the NR community regularly publishes catalogs of waveforms [9, 52–54]. I have contributed a range of new aligned-spin simulations using the `SpEC` code [55] to the SXS catalog [9]. Extending the parameter space of available NR simulations contributes directly to the capabilities of the waveform modeling community, and hence to the detection and source-characterization pipelines of current and future gravitational-wave observatories.

[41]: Antonelli et al. (2019), *Energetics of two-body Hamiltonians in post-Minkowskian gravity*
[42]: Antonelli et al. (2020), *Quasicircular inspirals and plunges from nonspinning effective-one-body Hamiltonians with gravitational self-force information*

[43]: Barausse and Buonanno (2010), *An Improved effective-one-body Hamiltonian for spinning black-hole binaries*
[44]: Taracchini et al. (2014), *Effective-one-body model for black-hole binaries with generic mass ratios and spins*
[45]: Damour and Nagar (2014), *A new analytic representation of the ringdown waveform of coalescing spinning black hole binaries*
[46]: Cotesta et al. (2018), *Enriching the Symphony of Gravitational Waves from Binary Black Holes by Tuning Higher Harmonics*
[47]: Khan et al. (2016), *Frequency-domain gravitational waves from nonprecessing black-hole binaries. II. A phenomenological model for the advanced detector era*
[48]: Blackman et al. (2017), *Numerical relativity waveform surrogate model for generically precessing binary black hole mergers*
[49]: Varma et al. (2019), *Surrogate model of hybridized numerical relativity binary black hole waveforms*
[50]: Yoo et al. (2022), *Targeted large mass ratio numerical relativity surrogate waveform model for GW190814*

[9]: SXS (2019), *The SXS collaboration catalog of binary black hole simulations*
[52]: Dietrich et al. (2018), *CoRe database of binary neutron star merger waveforms*
[53]: Healy and Lousto (2022), *The Fourth RIT binary black hole simulations catalog: Extension to Eccentric Orbits*
[54]: Jani et al. (2016), *Georgia Tech Catalog of Gravitational Waveforms*

1.1.5 Multi-messenger astrophysics

Gravitational waves join the array of observable signals that we receive from the most violent events in our Universe, complementing observations of light across the electromagnetic spectrum, neutrinos, and high-energy massive cosmic particles [56]. Gravitational waves are uniquely positioned in this mix because they carry information from the strong-gravity regime, which is often obscured by absorption and scattering effects in optical channels. For the potential that the emerging fields of multimessenger astronomy and astrophysics hold, they have been identified as key research areas for the years to come [57–59].

For example, we expect that binary neutron star (BNS) mergers can be the source of short gamma-ray bursts (GRBs) alongside their emission of gravitational waves. This hypothesis was confirmed by the landmark multimessenger observation of the GW signal `GW170817` and the coincident GRB `170817A` on Aug 17, 2017 [60–62]. This single GW-multimessenger event has already enabled a test of the propagation speed of gravitational radiation, has ruled out swathes of modified gravity theories, enabled a

[56]: Mészáros et al. (2019), *Multi-Messenger Astrophysics*

[57]: National Academies of Sciences, Engineering, and Medicine (2021), *Pathways to Discovery in Astronomy and Astrophysics for the 2020s (decadal survey)*
[58]: Foley et al. (2019), *Gravity and Light: Combining Gravitational Wave and Electromagnetic Observations in the 2020s*
[59]: Kalogera et al. (2021), *The Next Generation Global Gravitational Wave Observatory: The Science Book*

[60]: LIGO, Virgo (2017), *GW170817: Observation of Gravitational Waves from a Binary Neutron Star Inspiral*
[61]: LIGO Scientific, Virgo, Fermi-GBM, INTEGRAL (2017), *Gravitational Waves and Gamma-rays from a Binary Neutron Star Merger: GW170817 and GRB 170817A*
[62]: LIGO, Virgo, et al. (2017), *Multi-messenger Observations of a Binary Neutron Star Merger*

new way of measuring the Hubble constant, established BNS mergers as a likely source of heavy elements in the Universe by r-process nucleosynthesis, and constrained the equation of state for matter at supranuclear densities. See Dietrich, Hinderer, and Samajdar [63] for a recent review focused on the GW signal from BNS mergers, and references therein. Other possible, but as yet unobserved, GW-multimessenger sources include tidal disruption events of stars by massive black holes, supernovae, and accreting supermassive BBH systems. See Mészáros et al. [56] for a recent review.

[63]: Dietrich, Hinderer, and Samajdar (2021), *Interpreting Binary Neutron Star Mergers*

[56]: Mészáros et al. (2019), *Multi-Messenger Astrophysics*

Numerical relativity simulations are at the forefront of studies that extract, analyze, and interpret multimessenger signals involving gravitational waves. To make detections, the matched-filtering and parameter-estimation techniques detailed above require accurate gravitational waveforms of BNS and BHNS mergers informed by numerical simulations [63]. Once a detection is made, numerical simulations probe how accurately we understand the multitude of physics at play in these complex astrophysical scenarios. The regimes involved are often inaccessible for laboratories on Earth, in particular when they take place in strong gravity, promoting numerical relativity to an essential tool in multimessenger astrophysics. The multiphysics nature of these astrophysical events makes them particularly challenging to simulate numerically, as I will detail in Section 1.1.6 below. Much of the motivation for the work in this thesis is to gear up our computational capabilities for the multimessenger era.

1.1.6 Challenges in numerical relativity

The plan for the future of numerical relativity is rather clear, though no less challenging. One obvious objective is to serve the next generation of gravitational wave observatories that are in development worldwide [59], such as the ground-based Cosmic Explorer [64] and Einstein Telescope [65] projects, and the space-based LISA [66] mission.[14] To keep up with the increase in sensitivity expected from these "3G" detectors, Pürrer and Haster [67] find that waveforms extracted from NR simulations must increase in accuracy by about a factor of ten, and also extend in both length and parameter space coverage. Ferguson et al. [68] find that NR codes must improve significantly to meet this challenge, and suggest higher-order discretization schemes as a possible path forward. Pürrer and Haster [67] also suggest new approaches to parallelization and to waveform extraction. All three aspects are being developed as part of the new `SpECTRE` numerical relativity code [10], which is the main focus of this thesis and introduced in Section 1.3.6. Most of my own efforts have focused on the higher-order discretization and parallelization aspects, specifically for the initial data solver of the code, which I will describe in detail in this thesis. A novel waveform extraction procedure based on the Cauchy-characteristic extraction (CCE) method is the subject of one of my co-author publications, Ref. [6].

[59]: Kalogera et al. (2021), *The Next Generation Global Gravitational Wave Observatory: The Science Book*

[64]: Evans et al. (2021), *A Horizon Study for Cosmic Explorer: Science, Observatories, and Community*

[65]: Maggiore et al. (2020), *Science Case for the Einstein Telescope*

[66]: eLISA (2013), *The Gravitational Universe*

14: The Cosmic Explorer Horizon Study, Ref. [64], uses one of my GW visualizations on the cover. Appendix A discusses these visualizations.

[67]: Pürrer and Haster (2020), *Gravitational waveform accuracy requirements for future ground-based detectors*

[68]: Ferguson et al. (2021), *Assessing the readiness of numerical relativity for LISA and 3G detectors*

[10]: SpECTRE, spectre-code.org

[6]: Moxon et al. (2021), *The SpECTRE Cauchy-characteristic evolution system for rapid, precise waveform extraction*

A key challenge of simulating multimessenger scenarios, such as those outlined in Section 1.1.5 above, is their multiphysics nature. Simulations involve the general-relativistic magnetohydrodynamic (GRMHD) effects of charged and self-gravitating matter, the nuclear physics of matter at or above nuclear density encoded in the equation of state of neutron stars, and the radiation transport of high-energy neutrinos, to name just a few

areas that are at the front of contemporary research. Turbulent mechanisms and instabilities, such as the magnetohydrodynamic instability (MRI) and the Kelvin-Helmholtz instability, can have large-scale effects and require a high dynamic range of resolutions within a simulation. Shocks and other discontinuities, such as neutron star surfaces and phase transitions in the equation of state, also require careful numerical treatment. A primary objective of the `SpECTRE` code is to simulate such multiphysics scenarios. For example, one of my co-author publications, Ref. [7], studies high-resolution shock capturing (HRSC) methods ("limiters") for a high-order numerical discretization scheme. See also Most, Papenfort, and Rezzolla [69] for recent progress on high-order numerical methods for GRMHD simulations, Foucart et al. [70] for neutrino transport, and Dietrich, Hinderer, and Samajdar [63], and references therein, for a discussion of numerical challenges in BNS mergers.

[7]: Deppe et al. (2021), *Simulating magnetized neutron stars with discontinuous Galerkin methods*

[69]: Most, Papenfort, and Rezzolla (2019), *Beyond second-order convergence in simulations of magnetized binary neutron stars with realistic microphysics*

[70]: Foucart et al. (2020), *Monte-Carlo neutrino transport in neutron star merger simulations*

[63]: Dietrich, Hinderer, and Samajdar (2021), *Interpreting Binary Neutron Star Mergers*

Alongside theoretical challenges to formulate numerical schemes stand computational challenges. We have access to rapidly increasing computing resources in the form of large and readily available supercomputers. Since the increase in speed of individual processors has largely stalled, modern supercomputers focus on increasing the sheer number of available compute cores per node, along with fast interconnects between nodes, hardware accelerators such as GPUs, and fast memory. Therefore, the capability to parallelize numerical simulations effectively across the processors of large computing clusters is a key challenge for modern numerical relativity codes. Parallelization is at the core of the new `SpECTRE` code and a major focus of this thesis, specifically in Chapter 3.

All numerical simulations begin with initial data that capture the physical scenario at hand, so advances in theory and technology for time evolution codes must be supported by sufficiently precise initial data. In fact, numerical effects aside, the result of a simulation is determined entirely by the initial data and boundary conditions (see also Section 1.2 below).[15] Therefore, to incorporate more detailed and more accurate microphysics in simulations also requires correspondingly detailed and accurate initial data.

15: Note that some recent developments employ stochastic numerical methods, such as Monte-Carlo neutrino transport [70]. These methods limit the determinism of the simulation to the numerical resolution of the stochastic method.

A classic example for the importance of initial data in BBH mergers are high-spin simulations, and the challenge to eliminate spurious gravitational radiation: straightforward "puncture" initial data [71] (Section 1.2.2) can represent black holes up to dimensionless spins of $\chi \lesssim 0.94$ [72, 73], and extensions to the formulation were required to represent higher spins [74–76] (Section 1.2.3). The challenge with binary initial data sets is that they must not only satisfy the Einstein equations, but also represent a quasiequilibrium inspiral of the two bodies. Initial deviations from quasiequilibrium will relax during the evolution and lead to spurious gravitational radiation ("junk radiation"). Early in the simulation, this radiation will either propagate out of the computational domain, carrying away some energy-momentum, or fall into the black holes, increasing their mass and hence decreasing their dimensionless spin. Junk radiation is also inherently difficult to resolve numerically because it typically has high frequency content, slowing down simulations early on or straining their adaptive mesh refinement schemes [9, 77]. Therefore, an ongoing quest in numerical relativity is generating initial data with minimal junk radiation. Recent approaches include numerical experiments with BBH initial data based on superposed Kerr solutions in various coordinate

[71]: Brandt and Brügmann (1997), *A Simple construction of initial data for multiple black holes*

[72]: Dain, Lousto, and Zlochower (2008), *Extra-Large Remnant Recoil Velocities and Spins from Near-Extremal-Bowen-York-Spin Black-Hole Binaries*

[73]: Dain, Lousto, and Takahashi (2002), *New conformally flat initial data for spinning black holes*

[74]: Pfeiffer (2005), *The Initial value problem in numerical relativity*

[75]: Lovelace et al. (2008), *Binary-black-hole initial data with nearly-extremal spins*

[76]: Ruchlin et al. (2017), *Puncture Initial Data for Black-Hole Binaries with High Spins and High Boosts*

[9]: SXS (2019), *The SXS collaboration catalog of binary black hole simulations*

[77]: Lovelace (2009), *Reducing spurious gravitational radiation in binary-black-hole simulations by using conformally curved initial data*

systems [78], and attempts to eliminate the dependence on Kerr-like backgrounds [79, 80], which are related to the line of work on the "waveless" initial data formulation (see Uryū et al. [81]).

Initial data generation for multiphysics scenarios, such as BNS or BHNS systems, is complicated by equilibrium conditions on the matter content (see Section 1.2.4, and the recent review by Kyutoku, Shibata, and Taniguchi [82]). Contemporary codes typically employ fixed-point iteration procedures that successively solve the gravity sector, the matter sector, and additional constraints in each iteration. These algorithms are strongly damped so they remain convergent, which slows them down significantly (see Section 4.4 for a discussion). Furthermore, initial data solvers for orbiting binaries with neutron stars are as yet limited to spinning, self-gravitating fluids in quasiequilibrium inspirals, with additional effects such as magnetic fields and neutrino radiation superimposed *a posteriori* [83, 84]. To seed numerical simulations for the multimessenger era, next-generation numerical codes such as `SpECTRE` need high-fidelity initial data solvers.

Initial data problems require numerical solutions to elliptic PDEs (Section 1.2), which pose computational challenges different to time evolution schemes. Since elliptic problems represent constraints on fields within a computational domain, they are inherently global: boundary conditions on one side of the domain determine the solution on the other side. Information must propagate across all grid points, making numerical algorithms to solve elliptic PDEs hard to scale with increasing resolution. Multigrid algorithms have proven successful to mitigate this issue and have been adopted by some numerical relativity codes [85–88], but often with restrictions such as Cartesian grids. The global nature of elliptic problems also makes algorithms hard to parallelize on computing clusters. Recent approaches such as Papenfort et al. [89] employ a large number of processors, but no acceleration techniques such as a multigrid method. This thesis approaches the initial data problem in numerical relativity from a primarily computational perspective, developing a scalable and parallel elliptic solver for the `SpECTRE` code intended to seed our numerical simulations with initial data for years to come.

Historically, a core challenge in NR has been the development of numerically well-behaved formulations of the Einstein equations. This remains an active area of research in the quest to extend NR simulations to theories of gravity beyond GR. See the lines of work around Witek et al. [90] and Okounkova [91] for recent progress in this field. I will place no further focus on modified gravity theories in this thesis. However, note that the topic of well-defined numerical formulations is strongly connected to the classification of equations of motion in elliptic and hyperbolic PDEs, which I will discuss next. The initial data problem in numerical relativity beyond GR is also quite unexplored as yet.

1.2 Elliptic equations in numerical relativity

Elliptic equations are important in many areas of physics, including numerical relativity. They often constrain admissable field configurations at all times during an evolution. Before delving into the specifics of elliptic

[78]: Varma, Scheel, and Pfeiffer (2018), *Comparison of binary black hole initial data sets*

[79]: Tsokaros, Uryū, and Shapiro (2019), *Complete initial value spacetimes containing black holes in general relativity: Application to black hole-disk systems*

[80]: Grandclément and Nicoules (2022), *Boundary conditions for stationary black holes: Application to Kerr, Martínez-Troncoso-Zanelli, and hairy black holes*

[81]: Uryū et al. (2016), *New code for equilibriums and quasiequilibrium initial data of compact objects. III. Axisymmetric and triaxial rotating stars*

[82]: Kyutoku, Shibata, and Taniguchi (2021), *Coalescence of black hole–neutron star binaries*

[83]: Kiuchi et al. (2014), *High resolution numerical-relativity simulations for the merger of binary magnetized neutron stars*

[84]: Hayashi et al. (2021), *General-relativistic neutrino-radiation magnetohydrodynamics simulation of black hole-neutron star mergers for seconds*

[85]: Cheong, Lin, and Li (2020), *Gmunu: Toward multigrid based Einstein field equations solver for general-relativistic hydrodynamics simulations*

[86]: East, Ramazanoglu, and Pretorius (2012), *Conformal Thin-Sandwich Solver for Generic Initial Data*

[87]: Moldenhauer et al. (2014), *Initial data for binary neutron stars with adjustable eccentricity*

[88]: Ansorg, Brügmann, and Tichy (2004), *A Single-domain spectral method for black hole puncture data*

[89]: Papenfort et al. (2021), *New public code for initial data of unequal-mass, spinning compact-object binaries*

[90]: Witek et al. (2019), *Black holes and binary mergers in scalar Gauss-Bonnet gravity: scalar field dynamics*

[91]: Okounkova (2020), *Numerical relativity simulation of GW150914 in Einstein dilaton Gauss-Bonnet gravity*

PDEs in numerical relativity, I review the essentials of PDE classification with a focus on elliptic equations.

For example, the classic Maxwell equations for the electric field $\boldsymbol{E}$ and the magnetic field $\boldsymbol{B}$ split in a set of two constraint equations and two time-evolution equations,

$$\boldsymbol{\nabla} \cdot \boldsymbol{E} = 4\pi\rho, \qquad \boldsymbol{\nabla} \cdot \boldsymbol{B} = 0, \tag{1.33a}$$

$$\partial_t \boldsymbol{E} = \boldsymbol{\nabla} \times \boldsymbol{B} - 4\pi \boldsymbol{j}, \qquad \partial_t \boldsymbol{B} = -\boldsymbol{\nabla} \times \boldsymbol{E}, \tag{1.33b}$$

where ρ is the electric charge density and $\boldsymbol{j}$ is the current density. The split in constraint and evolution equations is particularly clear in Coulomb gauge, $\boldsymbol{\nabla} \cdot \boldsymbol{A} = 0$, when we employ the vector potential $\boldsymbol{A}$ for the magnetic field $\boldsymbol{B} = \boldsymbol{\nabla} \times \boldsymbol{A}$, and the scalar potential φ for the electric field $\boldsymbol{E} = -\boldsymbol{\nabla}\varphi - \partial_t \boldsymbol{A}$. Then, the Maxwell equations (1.33) reduce to

$$-\Delta\varphi = 4\pi\rho, \tag{1.34a}$$

$$-\Box \boldsymbol{A} = 4\pi \boldsymbol{j} - \partial_t \boldsymbol{\nabla}\varphi, \tag{1.34b}$$

where $\Delta = \partial_i \partial^i$ is the Laplace operator and $\Box = \left(\partial_t^2 - \Delta\right)$ is the d'Alembertian wave operator. The constraint (1.34a) is an elliptic equation that the electric potential φ must satisfy at all times. It complements the hyperbolic equation (1.34b) that governs the evolution of the vector potential $\boldsymbol{A}$ in time, for which φ is a source field [92].

[92]: Knapp, Walker, and Baumgarte (2002), *Illustrating Stability Properties of Numerical Relativity in Electrodynamics*

More generally, field theories in physics are typically governed by second-order partial differential equations (PDEs), and these PDEs are usually of elliptic, hyperbolic, or parabolic type. The prototypical elliptic equation is the Poisson equation,

$$-\Delta u = f(\boldsymbol{x}), \tag{1.35}$$

for the field $u(\boldsymbol{x})$ sourced by a function $f(\boldsymbol{x})$, such as the Maxwell constraint (1.34a). Elliptic, and specifically quasilinear Poisson-type equations, are ubiquitous in physics, appearing in Newtonian gravity, electrodynamics [Eq. (1.34a)], fluid dynamics [Eq. (1.63)], elasticity [Eq. (5.9)], steady-state heat diffusion, and, as we will encounter below, in general relativity [Eqs. (1.51) and (1.56)]. Mathematically, PDEs are classified by the properties of their principal symbol, which represents the highest-order derivatives in the equations. Specifically, a general linear PDE of second order has the form

$$A^{ij} \frac{\partial^2 u}{\partial x^i \partial x^j} + \text{lower-order terms} = 0, \tag{1.36}$$

where the symmetric coefficients A^{ij} represent the principal symbol. The principal symbol is directly related to the geometry and causal structure underlying the field theory, such as the metric of the spacetime manifold, and specifically its signature.[16] The PDE is elliptic when all eigenvalues of A^{ij} have the same sign, as is the case for the Poisson equation, Eq. (1.35). On the other hand, the principal coefficient matrix of the wave equation, Eq. (1.34b), has one eigenvalue of opposite sign, which is the defining property of a hyperbolic equation.

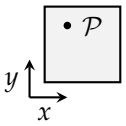


Figure 1.10: Elliptic PDEs determine the solution at any point $\mathcal{P}$ within a domain given enveloping boundary conditions.

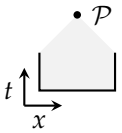


Figure 1.11: Hyperbolic PDEs determine the solution at a point $\mathcal{P}$ given initial data on a slice through its past characteristic cone. Boundary conditions are needed for incoming modes where the cone intersects the domain boundary.

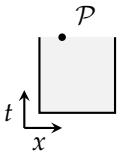


Figure 1.12: Parabolic PDEs determine the solution at a point $\mathcal{P}$ given initial data and enclosing boundary conditions.

Elliptic PDEs typically represent static constraint problems, such as the charge density inducing an electric potential by the Maxwell constraint (1.34a). They require boundary conditions enclosing a computa-

16: Incidentally, and largely unrelated to this thesis, the relation between causality and geometrodynamics is the subject of my co-author publication [13] on the gravitational closure framework.

tional domain to determine the field within, as illustrated in Fig. 1.10. On the other hand, hyperbolic PDEs typically represent causal time evolutions, where information propagates with finite *characteristic speeds* (Fig. 1.11). Hyperbolic PDEs require initial data on a spatial slice through the spacetime cones spanned by their characteristics. They also require boundary conditions where the characteristic cones intersect the boundary of the computational domain. Boundary conditions are only needed for the incoming modes of the propagating fields, i.e., those with negative characteristic speed orthogonal to the boundary. The analysis and numerical evolution of hyperbolic PDEs is a rich area of research. For the purpose of this thesis, it suffices to recognize that the initial data required for the evolution of hyperbolic PDEs in physics is often constrained by elliptic PDEs, as is the case for the simple example of the Maxwell equations in Eq. (1.34). Finally, parabolic PDEs are characterized by instantaneous propagation of information across the computational domain. They require initial data and boundary conditions enclosing the domain at every instant in time (Fig. 1.12).

1.2.1 Einstein constraints and evolution

In numerical relativity we study the Einstein equations, Eq. (1.1). They represent ten coupled, nonlinear, second-order PDEs for the metric tensor $g_{\mu\nu}$. For computational purposes it is essential to decompose the Einstein equations into a set of elliptic constraint equations and hyperbolic evolution equations, just like the Maxwell equations (1.34), so the PDEs have well-defined mathematical and numerical properties. Since much of this thesis focuses on numerical solutions to the elliptic Einstein constraint equations, I outline their derivation in this section. See Baumgarte and Shapiro [17] for a more detailed introduction.

[17]: Baumgarte and Shapiro (2010), *Numerical Relativity: Solving Einsteins Equations on the Computer*

To formulate an initial value problem for general relativity suitable for numerical evolution we first have to clarify notions of space and time. This is accomplished by the 3+1 decomposition of general relativity. We foliate the four-dimensional spacetime manifold into spacelike hypersurfaces Σ_t, each defined as a level surface of a scalar function t. The function t is later chosen as the time coordinate. Hypersurfaces are characterized by their timelike unit normal

$$n^\mu = -\alpha g^{\mu\nu}\nabla_\nu t, \qquad \alpha = \left(-g^{\mu\nu}\nabla_\mu t \nabla_\nu t\right)^{-1/2}, \tag{1.37}$$

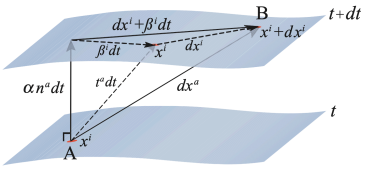


Figure 1.13: Geometry of two neighboring hypersurfaces (Fig. 2.4 in Baumgarte and Shapiro [17]).

where the *lapse* $\alpha > 0$ parametrizes the amount of proper time that elapses from one spatial hypersurface to the next, and n^μ is timelike by construction, $n^\mu n_\mu = -1$. Correspondingly, the *shift vector* β^μ is defined by

$$t^\mu = \alpha n^\mu + \beta^\mu, \tag{1.38}$$

where $t^\mu = (1, 0, 0, 0)$ connects points with the same spatial coordinates between hypersurfaces (see Fig. 1.13). The spacetime metric $g_{\mu\nu}$ induces a spatial metric $\gamma_{\mu\nu}$ on the hypersurfaces Σ_t,

$$\gamma_{\mu\nu} = g_{\mu\nu} + n_\mu n_\nu. \tag{1.39}$$

It defines a spatial covariant derivative $\mathcal{D}_\mu$ on Σ_t complete with Christoffel symbols and Ricci curvature, and also serves to project spacetime tensors

into the spatial hypersurfaces. The gradient of the timelike unit normal projected to Σ_t is the *extrinsic curvature*,

$$K_{\mu\nu} = -\gamma_\mu{}^\rho \gamma_\nu{}^\sigma \nabla_\rho n_\sigma \tag{1.40a}$$

$$= -\frac{1}{2}\mathcal{L}_{\boldsymbol{n}} \gamma_{\mu\nu}. \tag{1.40b}$$

It is a symmetric spatial tensor with the notion of a curvature because it quantifies the deviation of the normal vector from parallel transport on Σ_t. The extrinsic curvature can also be expressed as the Lie derivative of the spatial metric along the timelike unit normal n^μ, which bestows on it the notion of the "time derivative" of $\gamma_{\mu\nu}$.

The tensors $\gamma_{\mu\nu}$, $K_{\mu\nu}$, and β^μ are purely spatial, meaning their contractions with the timelike unit normal n^μ vanish. Therefore, the spatial components γ_{ij}, K_{ij}, and β^i suffice to describe these tensors when the level function t is chosen as coordinate time. In these coordinates, the unit normal vector is $n^\mu = \left(\alpha^{-1}, -\alpha^{-1}\beta^i\right)$ or $n_\mu = (-\alpha, 0, 0, 0)$, the spatial metric is $\gamma_{ij} = g_{ij}$, and the spacetime metric takes the form

$$\mathrm{d}s^2 = -\alpha^2 \,\mathrm{d}t^2 + \gamma_{ij}\left(\mathrm{d}x^i + \beta^i \,\mathrm{d}t\right)\left(\mathrm{d}x^j + \beta^j \,\mathrm{d}t\right) \tag{1.41}$$

in terms of the 3+1 quantities.

In the 3+1 decomposition, the Einstein equations (1.1) take the standard Arnowitt-Desner-Misner (ADM) form [93, 94],[17]

[93]: Arnowitt, Deser, and Misner (1962), *The dynamics of general relativity*
[94]: York (1979), *Kinematics and Dynamics of General Relativity*
17: Box 2.1 in Baumgarte and Shapiro [17]

$$R + K^2 - K_{ij}K^{ij} = 16\pi\rho_{\mathrm{H}} \tag{1.42a}$$

$$\mathcal{D}_j\left(K^{ij} - \gamma^{ij}K\right) = 8\pi S^i \tag{1.42b}$$

$$\partial_t \gamma_{ij} = -2\alpha K_{ij} + \mathcal{D}_i\beta_j + \mathcal{D}_j\beta_i \tag{1.42c}$$

$$\begin{aligned}\partial_t K_{ij} = {} & \alpha\left(R_{ij} - 2K_{ik}K^k{}_j + KK_{ij}\right) \\ & - \mathcal{D}_i\mathcal{D}_j\alpha - 8\pi\alpha\left(S_{ij} - \frac{1}{2}\gamma_{ij}\left(S - \rho_{\mathrm{H}}\right)\right) \\ & + \beta^k\partial_k K_{ij} + K_{ik}\partial_j\beta^k + K_{kj}\partial_i\beta^k.\end{aligned} \tag{1.42d}$$

Here, the Ricci scalar R in Eq. (1.42a) and the Ricci tensor R_{ij} in Eq. (1.42d) are defined with respect to the spatial metric γ_{ij}, and $K = \gamma^{ij}K_{ij}$ is the trace of the extrinsic curvature. The 3+1 matter sources in the ADM equations are the projections of the energy-momentum $T^{\mu\nu}$,

$$\rho_{\mathrm{H}} = n_\mu n_\nu T^{\mu\nu}, \tag{1.43a}$$

$$S^i = -\gamma^{ij} n^\mu T_{\mu j}, \tag{1.43b}$$

$$S_{ij} = \gamma_{i\mu}\gamma_{j\nu}T^{\mu\nu}, \tag{1.43c}$$

$$S = \gamma^{ij}S_{ij}. \tag{1.43d}$$

Similar to the Maxwell equations (1.33), the ADM equations (1.42) split into a set of time-independent constraints—the *Hamiltonian constraint* (1.42a) and the *momentum constraint* (1.42b)—and a set of evolution equations that conserve the constraints, Eqs. (1.42c) and (1.42d). Extending the similarities with the Maxwell equations, the ADM equations admit a gauge freedom in the choice of four degrees of freedom in the lapse and shift. The Hamiltonian and momentum constraints determine

another four degrees of freedom. The remaining four degrees of freedom in the spatial metric γ_{ij} and the extrinsic curvature K_{ij} define the physical scenario at hand. The challenge in constructing initial data is to make suitable choices for these remaining degrees of freedom, possibly in a favorable gauge, and then solve the constraint equations numerically to obtain valid data (γ_{ij}, K_{ij}) on a spatial hypersurface.

A conformal decomposition has proven a successful strategy to cast the Einstein constraints in a form suitable to make informed choices for the dynamic degrees of freedom. The idea is to expose transverse-traceless degrees of freedom that represent propagating gravitational waves (see Section 1.1.1), so they can be set to zero for quasiequilibrium scenarios. We decompose the spatial metric as [95]

[95]: Lichnerowicz (1944), *L'intégration des équations de la gravitation relativiste et la problème des n corps*

$$\gamma_{ij} = \psi^4 \bar{\gamma}_{ij}, \tag{1.44}$$

where $\psi > 0$ is the *conformal factor* and $\bar{\gamma}_{ij}$ is the *conformal metric*. We also decompose the extrinsic curvature as

$$K_{ij} = \psi^{-2} \bar{A}_{ij} + \frac{1}{3} \gamma_{ij} K, \tag{1.45}$$

where $\bar{A}_{ij}$ is the conformal traceless extrinsic curvature, and we treat the trace of the extrinsic curvature, K, as a conformal invariant. Under the conformal decomposition, the Hamiltonian and momentum constraints, Eqs. (1.42a) and (1.42b), become [94, 96] [18]

[94]: York (1979), *Kinematics and Dynamics of General Relativity*
[96]: O Murchadha and York (1974), *Gravitational energy*

18: Eqs. (3.37) and (3.38) in Baumgarte and Shapiro [17]

$$8\bar{\mathcal{D}}^2\psi - \psi\bar{R} - \frac{2}{3}\psi^5 K^2 + \psi^{-7}\bar{A}_{ij}\bar{A}^{ij} = -16\pi\psi^5\rho_{\mathrm{H}}, \tag{1.46a}$$

$$\bar{\mathcal{D}}_j\bar{A}^{ij} - \frac{2}{3}\psi^6\bar{\gamma}^{ij}\bar{\mathcal{D}}_j K = 8\pi\psi^{10}S^i, \tag{1.46b}$$

where $\bar{\mathcal{D}}_i$ is the covariant derivative associated with the conformal metric $\bar{\gamma}_{ij}$, and $\bar{R}$ is the corresponding conformal Ricci scalar. Before continuing along this route, I discuss a particularly straightforward way to furnish solutions to the constraint equations.

1.2.2 Puncture initial data

A popular approach to produce initial data for general-relativistic time evolutions involving black holes is the puncture method [71]. Given the complexity of the Einstein constraints, Eq. (1.46), the puncture method furnishes a remarkably straightforward method to construct initial data containing black holes. See Section 12.2 in Baumgarte and Shapiro [17] for a more detailed introduction to puncture initial data.

[71]: Brandt and Brügmann (1997), *A Simple construction of initial data for multiple black holes*

[17]: Baumgarte and Shapiro (2010), *Numerical Relativity: Solving Einsteins Equations on the Computer*

First, we simplify the constraints significantly by assuming conformal flatness and maximal slicing,

$$\bar{\gamma}_{ij} = \eta_{ij} \quad \text{and} \quad K = 0, \tag{1.47}$$

as well as vacuum, $T_{\mu\nu} = 0$. Under these assumptions, the Einstein

constraints (1.46) reduce to

$$\bar{\mathcal{D}}^2\psi = \frac{1}{8}\psi^{-7}\bar{A}_{ij}\bar{A}^{ij}, \tag{1.48a}$$

$$\bar{\mathcal{D}}_j\bar{A}^{ij} = 0, \tag{1.48b}$$

where $\mathcal{D}_i$ is the flat-space covariant derivative, i.e., simply ∂_i in Cartesian coordinates. The momentum constraint (1.48b) is now homogeneous. It is solved analytically by the Bowen-York extrinsic curvature, which can represent a black hole with linear momentum $\boldsymbol{P}$ and angular momentum $\boldsymbol{S}$ [97]. Since the momentum constraint (1.48b) is linear, Bowen-York solutions can be superposed to construct a solution for $\bar{A}^{ij}$ representing multiple black holes,[19]

$$\bar{A}^{ij} = \frac{3}{2}\sum_I \frac{1}{r_I^2}\left(2P_I^{(i}n_I^{j)} - (\eta^{ij} - n_I^i n_I^j)P_I^k n_I^k + \frac{4}{r_I}n_I^{(i}\epsilon^{j)kl}S_I^k n_I^l\right). \tag{1.49}$$

The black holes are parametrized by their positions $\boldsymbol{C}_I$, linear momenta $\boldsymbol{P}_I$, and spins $\boldsymbol{S}_I$. The quantity $r_I = \|\boldsymbol{x} - \boldsymbol{C}_I\|$ is the Euclidean coordinate distance to the Ith black hole, and $\boldsymbol{n}_I = (\boldsymbol{x} - \boldsymbol{C}_I)/r_I$ is the radial unit normal to the Ith black hole.

Now, only the Hamiltonian constraint (1.48a) remains to be solved. We decompose the conformal factor as [20]

$$\psi = 1 + \frac{1}{\alpha} + u \quad \text{with} \quad \frac{1}{\alpha} = \sum_I \frac{M_I}{r_I}, \tag{1.50}$$

where the parameters M_I are the *puncture masses*. The component $1 + 1/\alpha$ represents a superposition of black holes with neither momentum nor spin. It solves the homogeneous Hamiltonian constraint, $\bar{\mathcal{D}}^2\psi = 0$, and reduces to the Schwarzschild solution in isotropic coordinates for a single black hole. Therefore, the terms M_I/r_I enforce the existence of black holes at the positions $\boldsymbol{C}_I$. The additional *puncture field* u corrects for the momentum and spin of the black holes such that ψ solves the full inhomogeneous Hamiltonian constraint, Eq. (1.48a). With the decomposition (1.50), the Hamiltonian constraint reduces to a single nonlinear elliptic PDE for the field u, the *puncture equation*

$$-\bar{\mathcal{D}}^2 u = \beta\left(\alpha\left(1 + u\right) + 1\right)^{-7} \quad \text{with} \quad \beta = \frac{1}{8}\alpha^7\bar{A}_{ij}\bar{A}^{ij}. \tag{1.51}$$

Given a choice of black hole parameters $\{M_I, \boldsymbol{C}_I, \boldsymbol{P}_I, \boldsymbol{S}_I\}$, the puncture equation (1.51) is solved numerically for the field $u(\boldsymbol{x})$. Then, an admissible spacetime metric and extrinsic curvature can be assembled by retracing the series of decompositions. The quantities α and β are background fields that define the configuration of black holes. We supplement Eq. (1.51) with boundary conditions to impose asymptotic flatness far away from the black holes, $u \to 0$ as $r \to \infty$. Strategies to numerically solve nonlinear elliptic equations such as the puncture equation (1.51) are the subject of Section 1.3 and the remainder of this thesis.

The puncture approach has become the initial data method of choice for many numerical relativity codes, including two of the three first BBH merger simulations [21, 22].[21] Puncture initial data is particularly useful for codes that evolve the initial data with the Baumgarte-Shapiro-Shibata-

[97]: Bowen and York (1980), *Time asymmetric initial data for black holes and black hole collisions*

19: Eqs. (12.42) to (12.44) in Baumgarte and Shapiro [17]

20: See Eq. (12.50) in Baumgarte and Shapiro [17]. Note that α, and β below, should not be confused with the lapse and shift in this context.

[20]: Pretorius (2005), *Evolution of binary black hole spacetimes*

[21]: Campanelli et al. (2006), *Accurate evolutions of orbiting black-hole binaries without excision*

[22]: Baker et al. (2006), *Gravitational wave extraction from an inspiraling configuration of merging black holes*

21: The first simulation by Pretorius [20] employed initial data derived from scalar field gravitational collapse following Ref. [98].

[98]: Pretorius (2005), *Numerical relativity using a generalized harmonic decomposition*

Nakamura (BSSN) formulation of the Einstein evolution equations and with a moving puncture scheme. See Baumgarte and Shapiro [17] for an introduction to the BSSN formulation and to moving puncture schemes.

[76]: Ruchlin et al. (2017), *Puncture Initial Data for Black-Hole Binaries with High Spins and High Boosts*
[99]: Khamesra, Gracia-Linares, and Laguna (2021), *Black hole–neutron star binary mergers: the imprint of tidal deformations and debris*
[100]: Clark and Laguna (2016), *Bowen-York Type Initial Data for Binaries with Neutron Stars*
[101]: Kyutoku et al. (2021), *Reducing orbital eccentricity in initial data of black hole–neutron star binaries in the puncture framework*
[102]: Etienne et al. (2007), *Filling the holes: Evolving excised binary black hole initial data with puncture techniques*
[103]: Chaurasia, Dietrich, and Rosswog (2021), *Black hole-neutron star simulations with the BAM code: First tests and simulations*

However, the puncture method provides no immediate mechanism to impose a notion of quasiequilibrium on binary initial data. The puncture masses, centers, linear momenta, and spins must be chosen appropriately, often based on energy arguments. Puncture initial data is also difficult to combine with black-hole excision schemes, since it provides no control over the initial deformation and motion of the black hole horizons. Furthermore, extensions to the puncture method that relax the assumption of conformal flatness are needed to produce initial data with high spins, since Kerr black holes admit no conformally flat slices [76] (see the discussion on junk radiation in Section 1.1.6). Other extensions to the puncture method incorporate fluid sources to accomodate BHNS initial data. For instance, Khamesra, Gracia-Linares, and Laguna [99] employ the formulation by Clark and Laguna [100], choosing spherically symmetric fluid sources so the momentum constraint can still be solved analytically. Kyutoku et al. [101] mix the puncture approach with the XCTS method detailed below to solve for BHNS initial data. Another line of work explored by Etienne et al. [102] and recently picked up by Chaurasia, Dietrich, and Rosswog [103] use XCTS initial data directly, and resort to "stuffing the black hole" with fiducial data to accomodate their moving puncture BHNS simulations.

1.2.3 XCTS initial data

[104]: York (1999), *Conformal 'thin sandwich' data for the initial-value problem of general relativity*
[105]: Pfeiffer and York (2003), *Extrinsic curvature and the Einstein constraints*

This thesis is concerned primarily with the extended conformal thin sandwich (XCTS) [104, 105] formulation of the Einstein constraint equations. See also Sections 3.3 and 12.3 in Baumgarte and Shapiro [17] for an introduction to XCTS initial data.

We pick up the derivation of the XCTS system at the ADM formulation of the Einstein constraints, Eq. (1.46). First, we conformally transform the evolution equation for the spatial metric, Eq. (1.42c), which is essentially the definition of the extrinsic curvature. To this end, we define the time derivative of the conformal metric,

$$\bar{u}_{ij} \equiv \partial_t \bar{\gamma}_{ij} \quad \text{and} \quad \bar{\gamma}^{ij} \bar{u}_{ij} \equiv 0. \tag{1.52}$$

We also define the *longitudinal operator*,

$$\left(\bar{L}\beta\right)^{ij} = 2\bar{\nabla}^{(i}\beta^{j)} - \frac{2}{3}\bar{\gamma}^{ij}\bar{\nabla}_k\beta^k, \tag{1.53}$$

which conformally transforms as $(LV)^{ij} = \psi^{-4}\left(\bar{L}V\right)^{ij}$. Then, the evolution equation for the spatial metric, Eq. (1.42c), takes the form [22]

22: Eq. (3.101) in Baumgarte and Shapiro [17]

$$\bar{A}^{ij} = \frac{\psi^6}{2\alpha}\left(\left(\bar{L}\beta\right)^{ij} - \bar{u}^{ij}\right), \tag{1.54}$$

where we used the conformal decomposition of the extrinsic curvature, Eq. (1.45).

Second, we take the trace of the evolution equation for the extrinsic curvature, Eq. (1.42d), to find [23]

23: Eq. (2.137) in Baumgarte and Shapiro [17]

$$\mathcal{D}^2\alpha = -\partial_t K + \alpha\left(K_{ij}K^{ij} + 4\pi\left(\rho_{\mathrm{H}} + S\right)\right) + \beta^i\mathcal{D}_i K. \tag{1.55}$$

This equation furnishes a condition on the lapse given a choice of $\partial_t K$, which can be conveniently set to zero to construct quasiequilibrium initial data.

Finally, we assemble the Hamiltonian constraint (1.46a), the lapse equation (1.55), and the momentum constraint (1.46b), to obtain the XCTS equations [74, 105],[24]

$$\bar{\mathcal{D}}^2\psi = \frac{1}{8}\psi\bar{R} + \frac{1}{12}\psi^5 K^2 - \frac{1}{8}\psi^{-7}\bar{A}_{ij}\bar{A}^{ij} - 2\pi\psi^5\rho_{\mathrm{H}}, \tag{1.56a}$$

$$\bar{\mathcal{D}}^2(\alpha\psi) = \alpha\psi\left(\frac{7}{8}\psi^{-8}\bar{A}_{ij}\bar{A}^{ij} + \frac{5}{12}\psi^4 K^2 + \frac{1}{8}\bar{R} + 2\pi\psi^4\left(\rho_{\mathrm{H}} + 2S\right)\right) - \psi^5\partial_t K + \psi^5\beta^i\bar{\mathcal{D}}_i K, \tag{1.56b}$$

$$\bar{\mathcal{D}}_i(\bar{L}\beta)^{ij} = (\bar{L}\beta)^{ij}\bar{\mathcal{D}}_i\ln(\bar{\alpha}) + \bar{\alpha}\bar{\mathcal{D}}_i\left(\bar{\alpha}^{-1}\bar{u}^{ij}\right) + \frac{4}{3}\bar{\alpha}\psi^6\bar{\mathcal{D}}^j K + 16\pi\bar{\alpha}\psi^{10}S^j, \tag{1.56c}$$

where $\bar{\alpha} = \alpha\psi^{-6}$, and $\bar{A}^{ij}$ is given by Eq. (1.54). The Hamiltonian constraint (1.56a) remains an equation for the conformal factor ψ. For the lapse equation (1.56b) we have combined Eq. (1.55) with the Hamiltonian constraint (1.56a).[25] The momentum constraint (1.56c) is an equation for the shift β^i, where we have used Eq. (1.54).

The XCTS equations are solved for the conformal factor ψ, the product of lapse and conformal factor $\alpha\psi$, and the shift vector β^i together with the boundary conditions detailed below. The remaining quantities in the equations, i.e., the conformal metric $\bar{\gamma}_{ij}$, the trace of the extrinsic curvature K, their respective time derivatives $\bar{u}_{ij}$ and $\partial_t K$, the energy density ρ_{H}, the stress-energy trace S, and the momentum density S^i, are freely-specifiable fields that define the scenario at hand. Of particular importance is the conformal metric $\bar{\gamma}_{ij}$, which defines the background geometry, the covariant derivative $\bar{\mathcal{D}}_i$, the Ricci scalar $\bar{R}$, and the longitudinal operator.

Initial data slices constructed from solutions to the XCTS equations are widely used in numerical relativity. The formulation allows to impose a notion of quasiequilibrium on the initial data by the choice $\bar{u}_{ij} = 0$ and $\partial_t K = 0$. It also provides initial choices for the lapse and shift, relieving the numerical relativist from making choices for these variables. However, we must still impose a conformal metric $\bar{\gamma}_{ij}$ and a mean extrinsic curvature K. Common choices for these quantities are conformal flatness, $\bar{\gamma}_{ij} = \eta_{ij}$, and maximal slicing, $K = 0$, though these do not allow for high-spin black hole solutions as discussed in Section 1.1.6. Instead, the conformal background for BBH initial data is often constructed from superpositions of isolated Kerr solutions to achieve high spins [75, 78, 106]. The XCTS formulation is also prevalent for BNS and BHNS initial data. Many codes employ conformal flatness and maximal slicing for initial data with neutron stars [89, 107–110], though some codes also experiment with non-conformally-flat initial data [111, 112].

Note that the XCTS equations (1.56) are essentially two Poisson equations and one "minimal distortion" elasticity equation for the shift vector with

[74]: Pfeiffer (2005), *The Initial value problem in numerical relativity*
[105]: Pfeiffer and York (2003), *Extrinsic curvature and the Einstein constraints*

24: Box 3.3 in Baumgarte and Shapiro [17]

25: Note that $\mathcal{D}^2 f = \psi^{-4}\bar{\mathcal{D}}^2 f + 2\psi^{-4}\bar{\mathcal{D}}_i\ln\psi\bar{\mathcal{D}}^i f$ for a scalar f, and $\bar{\mathcal{D}}^2(\alpha\psi) = \alpha\bar{\mathcal{D}}^2\psi + \psi\bar{\mathcal{D}}^2\alpha + 2\bar{\mathcal{D}}_i\alpha\bar{\mathcal{D}}^i\psi$.

[75]: Lovelace et al. (2008), *Binary-black-hole initial data with nearly-extremal spins*
[78]: Varma, Scheel, and Pfeiffer (2018), *Comparison of binary black hole initial data sets*
[89]: Papenfort et al. (2021), *New public code for initial data of unequal-mass, spinning compact-object binaries*
[106]: Matzner, Huq, and Shoemaker (1999), *Initial data and coordinates for multiple black hole systems*
[107]: Dietrich et al. (2015), *Binary Neutron Stars with Generic Spin, Eccentricity, Mass ratio, and Compactness - Quasi-equilibrium Sequences and First Evolutions*
[108]: Rashti et al. (2021), *Elliptica: a new pseudo-spectral code for the construction of initial data*
[109]: Tacik et al. (2015), *Binary Neutron Stars with Arbitrary Spins in Numerical Relativity*
[110]: Tsokaros, Uryū, and Rezzolla (2015), *New code for quasiequilibrium initial data of binary neutron stars: Corotating, irrotational, and slowly spinning systems*
[111]: Uryū et al. (2009), *Non-conformally flat initial data for binary compact objects*
[112]: Tacik et al. (2016), *Initial data for black hole–neutron star binaries, with rotating stars*

coupled, nonlinear sources on a curved manifold. In this analogy, the longitudinal operator plays the role of the elastic constitutive relation that connects the symmetric "shift strain" $\bar{\mathcal{D}}_{(i}\beta_{j)}$ with the "stress" $(\bar{L}\beta)^{ij}$ of which we take the divergence in the momentum constraint (1.56c). This particular constitutive relation is equivalent to an isotropic and homogeneous material [Eq. (5.10)] with bulk modulus $K = 0$ (not to be confused with the extrinsic curvature trace K in this context) and shear modulus $\mu = 1$. Chapter 5 explores this relationship further.

Apparent-horizon boundary conditions

Black holes in the XCTS formalism are represented by excised regions of the computational domain. These regions are enclosed by boundary conditions that impose the excision surface is an *apparent horizon* in quasiequilibrium [113], so it will always lie within the event horizon of the spacetime. See Section 12.3.2 in Baumgarte and Shapiro [17] for a more detailed introduction to quasiequilibrium apparent-horizon boundary conditions.

[113]: Cook and Pfeiffer (2004), *Excision boundary conditions for black hole initial data*

An apparent horizon (AH) is a two-dimensional spatial surface that is characterized by the condition

$$\Theta = 0, \tag{1.57}$$

where Θ is the expansion of outgoing null geodesics on the surface,[26]

$$\Theta = \frac{1}{\sqrt{2}} m^{ij} \left(\mathcal{D}_i s_j - K_{ij} \right). \tag{1.58}$$

26: Eq. (7.29) in Baumgarte and Shapiro [17]

Here, s_i is the spatial unit normal to the apparent horizon, and $m_{ij} = \gamma_{ij} - s_i s_j$ is the induced metric on the surface. The condition (1.57) defines a *marginally outer-trapped surface* (MOTS), and an apparent horizon is the outermost of such surfaces. To impose the excision surface is not only a MOTS, but also in quasiequilibrium and remaining so initially in the chosen coordinates, we can impose the boundary conditions [113]

$$\begin{aligned} \bar{s}^k \mathcal{D}_k \psi = & -\frac{\psi^3}{8\alpha} \bar{s}_i \bar{s}_j \left((\bar{L}\beta)^{ij} - \bar{u}^{ij} \right) \\ & - \frac{\psi}{4} \bar{m}^{ij} \bar{\mathcal{D}}_i \bar{s}_j + \frac{1}{6} K \psi^3, \end{aligned} \tag{1.59a}$$

$$\beta^i = \frac{\alpha}{\psi^2} \bar{s}^i - \epsilon_{ijk} \Omega^j_r x^k \tag{1.59b}$$

on the XCTS variables, where $\bar{s}_i = \psi^{-2} s_i$ is the conformal unit normal to the surface, and $\bar{m}_{ij} = \bar{\gamma}_{ij} - \bar{s}_i \bar{s}_j$ is the conformal surface metric. The rotation parameters $\mathbf{\Omega}_r$ in Eq. (1.59b) endow the apparent horizon with a tangential shift-component. They twist the time vector t^μ, Eq. (1.38), and hence induce spin. The AH boundary conditions do not constrain the lapse, so we are free to choose another boundary condition for the lapse [113]. I will explore AH boundary conditions further in Chapter 4.

1.2.4 Fluid sources

When we simulate a spacetime with matter, the constraint equations are sourced by the energy-momentum $T^{\mu\nu}$ as given by Eq. (1.43). Matter involved in astrophysical scenarios, such as neutron stars, is typically modeled as a perfect fluid,[27]

$$T^{\mu\nu} = (\rho + P)\, u^\mu u^\nu + P g^{\mu\nu}, \tag{1.60}$$

where $\rho = \rho_0 (1 + \epsilon)$ is the total mass-energy density, ρ_0 is the rest-mass density, ϵ is the specific internal energy density, P is the pressure, and u^μ is the fluid four-velocity. We also define the specific enthalpy $h = 1+\epsilon+P/\rho_0$. The fluid must satisfy conservation of energy-momentum,

$$\nabla_\mu T^{\mu\nu} = 0, \tag{1.61}$$

and conservation of rest mass,

$$\nabla_\mu (\rho_0 u^\mu) = 0. \tag{1.62}$$

The fluid equations of motion are closed by an equation of state (EOS), such as $P(\rho_0)$. When we construct initial data involving matter we solve the hydrodynamic equations alongside the Einstein constraints to determine the fluid variables, as well as the spacetime metric. The matter and gravity sectors are coupled through the metric involved in the fluid equations, and the fluid sourcing the Einstein constraints. Therefore, we end up with a larger system of coupled equations that encompass both matter and gravity variables.

For example, in the important case of an irrotational neutron-star binary the fluid equations reduce to the elliptic PDE [28]

$$\mathcal{D}^2\Phi - \mathcal{D}_i\left(\Lambda B^i\right) = -\left(\mathcal{D}^i\Phi - \Lambda B^i\right)\mathcal{D}_i \ln\frac{\alpha\rho_0}{h} \tag{1.63}$$

for the velocity potential Φ defined by

$$h u_\mu =: \nabla_\mu \Phi, \tag{1.64}$$

and the algebraic equation

$$h^2 = \alpha^2\Lambda^2 - \mathcal{D}_i\Phi\mathcal{D}^i\Phi \tag{1.65}$$

for the specific enthalpy h. Here, $\Lambda = \alpha^{-2}\left(C + B^i\mathcal{D}_i\Phi\right)$, C is a constant of integration, $B^\mu = \beta^\mu + \Omega\varphi^\mu = \xi^\mu_{\rm hel} - \alpha n^\mu$ is the rotational shift, $\xi^\mu_{\rm hel} = t^\mu + \Omega\varphi^\mu$ is a helical Killing vector, t^μ is given by Eq. (1.38), and φ^μ describes orbital rotation with angular velocity Ω around the axis $\hat{z}$, so $\varphi^i = \epsilon^{ijk}\hat{z}_j x_k$ in Cartesian coordinates. The elliptic velocity potential equation (1.63) is supplemented by the regularity condition

$$\left(\mathcal{D}^i\Phi - \Lambda B^i\right)\mathcal{D}_i\rho_0\Big|_{\rm surface} = 0 \tag{1.66}$$

at the surface of the neutron stars where $h = 1$. I will not solve Eq. (1.63) numerically in this thesis, but note that it is a primary target for future applications of the computational methods developed here.

27: See, e.g., Rezzolla and Zanotti [114] for an introduction to relativistic hydrodynamics, or Baumgarte and Shapiro [17] for a summary.

[17]: Baumgarte and Shapiro (2010), *Numerical Relativity: Solving Einsteins Equations on the Computer*
[114]: Rezzolla and Zanotti (2013), *Relativistic Hydrodynamics*

28: Eq. (15.78) in Baumgarte and Shapiro [17], Eq. (4.27) in Moldenhauer et al. [87], and Eq. (A74) in Uryū et al. [111].

[17]: Baumgarte and Shapiro (2010), *Numerical Relativity: Solving Einsteins Equations on the Computer*
[87]: Moldenhauer et al. (2014), *Initial data for binary neutron stars with adjustable eccentricity*
[111]: Uryū et al. (2009), *Non-conformally flat initial data for binary compact objects*

1.2.5 More elliptic equations in numerical relativity

(1.55): $\mathcal{D}^2\alpha = -\partial_t K + \alpha\left(K_{ij}K^{ij} + 4\pi\left(\rho_{\mathrm{H}} + S\right)\right) + \beta^i\mathcal{D}_i K$

29: Eq. 4.13 in Baumgarte and Shapiro [17]

Elliptic equations arise not only in initial data problems. For example, in Eq. (1.55) we have taken the trace of the evolution equation for the extrinsic curvature to find an elliptic equation for the lapse. When we impose maximal slicing, $K = 0 = \partial_t K$, Eq. (1.55) reduces to [29]

$$\mathcal{D}^2\alpha = \alpha\big(R - 4\pi\left(3\rho_{\mathrm{H}} - S\right)\big). \tag{1.67}$$

To retain maximal slicing throughout a simulation, we could solve Eq. (1.67) alongside the evolution to control the lapse. This is not typically done because the available elliptic solver algorithms in numerical relativity have traditionally been too computationally expensive to make this strategy worthwile. Instead, gauge drivers are sometimes used that relax the lapse towards maximal slicing (K-driver) or towards other choices such as "1+log" slicing. Similarly, an elliptic equation similar to the XCTS equation for the shift, Eq. (1.56c), can be solved alongside an evolution to retain a "minimal distortion condition", but relaxation schemes such as a Gamma-driver are more commonly employed.[30]

30: See Box 4.1 in Baumgarte and Shapiro [17] and surrounding discussion.

(1.58): $\Theta = \frac{1}{\sqrt{2}} m^{ij}\left(\mathcal{D}_i s_j - K_{ij}\right)$

31: See Section 7.3 in Baumgarte and Shapiro [17].

Furthermore, the apparent-horizon condition (1.58) is an elliptic PDE when formulated in terms of a level function.[31] However, numerical relativity codes typically use a parabolic relaxation method to find apparent horizons throughout an evolution in an artificial time coordinate, such as the algorithm developed by Gundlach [115].

[115]: Gundlach (1998), *Pseudospectral apparent horizon finders: An Efficient new algorithm*

Recently, *constrained evolution* schemes for the Einstein equations have enjoyed renewed interest [85]. Fully constrained schemes only evolve the two dynamic gravitational-wave degrees of freedom using hyperbolic wave equations, and resort to elliptic equations to constrain all other degrees of freedom throughout the evolution [116]. This strategy can lead to exceptionally stable evolutions since it avoids the growth of constraint modes due to numerical error, but has seen little use in the recent development of numerical relativity, largely due to the computational cost associated with solving the elliptic constraint equations. Instead, contemporary codes typically employ hyperbolic *constraint damping* methods to stabilize evolutions [117, 118].

[85]: Cheong, Lin, and Li (2020), *Gmunu: Toward multigrid based Einstein field equations solver for general-relativistic hydrodynamics simulations*

[116]: Bonazzola et al. (2004), *A Constrained scheme for Einstein equations based on Dirac gauge and spherical coordinates*

[117]: Gundlach et al. (2005), *Constraint damping in the Z4 formulation and harmonic gauge*

[118]: Lindblom et al. (2006), *A New generalized harmonic evolution system*

The idea of constrained evolution can also be applied to perform *divergence cleaning*. For instance, in magnetohydrodynamic simulations care must be taken to avoid the growth of magnetic monopoles due to accumulation of numerical error, violating the Maxwell constraint $\boldsymbol{\nabla}\cdot\boldsymbol{B} = 0$, Eq. (1.34a). Many codes employ variations of hyperbolic divergence cleaning, propagating violations of the Maxwell constraint out of the computational domain using an additional scalar field [119]. Recently, Cheong et al. [120] have reported progress on elliptic divergence cleaning, where they solve the Maxwell constraint alongside the evolution.

[119]: Dedner et al. (2002), *Hyperbolic Divergence Cleaning for the MHD Equations*

[120]: Cheong et al. (2021), *An extension of Gmunu: General-relativistic resistive magnetohydrodynamics based on staggered-meshed constrained transport with elliptic cleaning*

A related concept where elliptic equations are prominent is implicit-explicit (IMEX) time stepping [121]. It is a strategy to accomodate "stiff" terms in evolution equations, meaning their relaxation time is small compared to the characteristic speeds of the remaining system. In such cases the evolution must resort to excessively small (explicit) time steps. Instead, IMEX methods treat the stiff sector of the equations implicitly, solving algebraic equations or elliptic PDEs alongside the evolution for the effect of the stiff sector. In numerical relativity, IMEX methods have

[121]: Pareschi and Russo (2005), *Implicit-explicit runge-kutta schemes and applications to hyperbolic systems with relaxation*

been explored by Lau, Lovelace, and Pfeiffer [122] for the gravity sector, and by Cheong et al. [120] and Ripperda et al. [123], and references therein for GRMHD simulations.

[122]: Lau, Lovelace, and Pfeiffer (2011), *Implicit-explicit (IMEX) evolution of single black holes*
[120]: Cheong et al. (2021), *An extension of Gmunu: General-relativistic resistive magnetohydrodynamics based on staggered-meshed constrained transport with elliptic cleaning*
[123]: Ripperda et al. (2019), *General relativistic resistive magnetohydrodynamics with robust primitive variable recovery for accretion disk simulations*

1.3 Numerical methods to solve elliptic equations

The plethora of elliptic PDEs that we have identified thus far can be solved only numerically in many interesting scenarios, particularly in the absence of symmetries or other simplifying assumptions. For instance, while the analytic Schwarzschild and Kerr solutions to the Einstein equations furnish essential insights into the physics of isolated spherically-symmetric and axisymmetric black holes, and represent impressive feats by themselves, numerical solutions are necessary to obtain nonperturbative solutions to the Einstein constraints representing orbiting black holes. The addition of astrophysical effects, such as magnetohydrodynamic matter with realistic equations of state, move the solution to the respective PDEs even further into the realm of computational physics.

PDEs are characterized by derivatives that couple degrees of freedom across multiple dimensions. This distinguishes them from ordinary differential equations (ODEs) that can be integrated along a single variable with considerably less computational effort (though numeric ODE integration is a rich research area as well, and I touch upon it when solving spherically symmetric stars in Section 4.3.1). In this section, I introduce the essential ideas and most prevalent numerical schemes for solving elliptic PDEs on a computer. Introductions to PDEs focused on computational physics can be found in Press et al. [124], Chapter 20, and Baumgarte and Shapiro [17], Chapter 6.

[124]: Press et al. (2007), *Numerical Recipes*
[17]: Baumgarte and Shapiro (2010), *Numerical Relativity: Solving Einsteins Equations on the Computer*

Elliptic PDEs, in contrast to hyperbolic and parabolic PDEs, represent static configurations with no notion of time. Instead of evolving data forward, numerical algorithms are tasked with finding an admissable configuration "everywhere at once". For instance, the Poisson equation (1.35) represents a condition on the field $u(\boldsymbol{x})$ given a fixed source $f(\boldsymbol{x})$ on a computational domain, as well as boundary conditions. The numerical methods outlined below find the solution $u(\boldsymbol{x})$ up to some numerical error. They discretize the field values on a computational grid so the elliptic equations take the form of a matrix equation,

(1.35): $-\Delta u = f(\boldsymbol{x})$

$$\mathcal{A}\underline{u} = \underline{b}, \tag{1.68}$$

where $\underline{u}$ represents the field values at all grid points, $\underline{b}$ represents the fixed sources at all grid points, and $\mathcal{A}$ is a matrix that represents the discrete elliptic operator (such as the Laplacian Δ in the Poisson equation). The matrix equation can then be inverted numerically to obtain the discrete solution $\underline{u}$ on the computational grid,

$$\underline{u} = \mathcal{A}^{-1}\underline{b}. \tag{1.69}$$

Nonlinear elliptic equations are typically linearized to obtain a matrix equation and then solved iteratively. In the remainder of this section I will outline some common computational methods to discretize and

invert elliptic PDEs. Then, looking ahead to the main subject of this thesis, Chapter 2 develops a discontinuous Galerkin scheme to construct the matrix $\mathcal{A}$ for a wide range of elliptic problems in numerical relativity, and Chapter 3 develops algorithms to invert the matrix on supercomputers.

1.3.1 Finite difference and finite volume methods

Finite difference (FD) methods discretize fields by their value at grid points. The computational domain Ω is typically covered by a regular and uniformly-spaced grid (Fig. 1.14). For instance, in two dimensions we might choose a grid spacing Δx along the direction x, and a spacing Δy along the direction y, so the grid points lie at

$$\boldsymbol{x}_p = \boldsymbol{x}_0 + \begin{pmatrix} p_x \Delta x \\ p_y \Delta y, \end{pmatrix} \tag{1.70}$$

where the index $p = (p_x, p_y)$ enumerates the grid points in both directions. The regular grid has a total of $N_{\text{points}} = N_x N_y$ grid points, where N_x and N_y count the number of grid points in each of the two dimensions. We denote the value of a scalar field $u(\boldsymbol{x})$ at the grid points $\boldsymbol{x}_p$ as

$$u_p = u(\boldsymbol{x}_p) \quad \text{and} \quad \underline{u} = \left(u_1, \ldots, u_{N_{\text{points}}}\right), \tag{1.71}$$

where $\underline{u}$ denotes the set of all scalar field values on the grid. With this discretization of the field, we can obtain a discretization of the derivative $\partial_i u$ in both directions of the grid by a simple Taylor expansion,

$$u_{(p_x \pm 1, p_y)} = u(x_p \pm \Delta x, y_p) = u_p \pm \Delta x \left(\partial_x u\right)_p + \frac{\Delta x^2}{2} \left(\partial_x^2 u\right)_p + \mathcal{O}(\Delta x^3), \tag{1.72a}$$

$$u_{(p_x, p_y \pm 1)} = u(x_p, y_p \pm \Delta y) = u_p \pm \Delta y \left(\partial_y u\right)_p + \frac{\Delta y^2}{2} \left(\partial_y^2 u\right)_p + \mathcal{O}(\Delta y^3). \tag{1.72b}$$

(1.35): $-\Delta u = f(\boldsymbol{x})$

To discretize the Laplacian $\Delta \equiv \partial_i \partial_i$ in the Poisson equation (1.35) we can take the sum of all four equations (1.72) so the first-derivative terms cancel. Then, the second-order FD approximation to the Poisson equation is

$$-\frac{u_{(p_x+1,p_y)} - 2u_p + u_{(p_x-1,p_y)}}{\Delta x^2} - \frac{u_{(p_x,p_y+1)} - 2u_p + u_{(p_x,p_y-1)}}{\Delta y^2} = f_p. \tag{1.73}$$

This equation has the form $\mathcal{A}\underline{u} = \underline{b}$, Eq. (1.68), since the left-hand side of Eq. (1.73), the *stencil*, defines a matrix-vector product. We can construct the matrix $\mathcal{A}$ explicitly by choosing an order in which we enumerate the values at the grid points collected in $\underline{u}$. Figure 1.15 illustrates the sparsity pattern of $\mathcal{A}$, i.e., its nonzero entries, where I have chosen the "column-major" order $p = p_x + N_x p_y$ (see Fig. 1.14). Boundary conditions can be imposed on the entries of the matrix corrsponding to the outermost grid points.

Finite volume (FV) methods are very similar to FD methods, and even equivalent for simple configurations. FV methods are based on flux

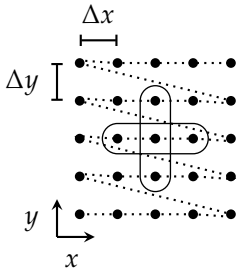


Figure 1.14: Regular and uniformly-spaced finite-difference grid. The circled grid points are coupled by the second-order FD stencil for the central point. The dotted line connects grid points in column-major order.

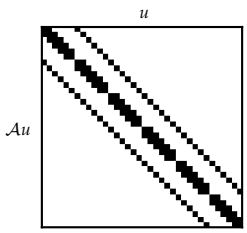


Figure 1.15: Sparsity pattern of the second-order finite difference discretization $\mathcal{A}$ of the Laplace operator for $N_x = N_y = 6$ grid points.

formulations of the PDEs. For instance, the Poisson equation (1.35) can be formulated as a divergence of the flux $v^i = \partial_i u$,

(1.35): $-\Delta u = f(\boldsymbol{x})$

$$-\partial_i v^i = f(\boldsymbol{x}). \tag{1.74}$$

Instead of evaluating the equation at a set of regularly spaced grid points (as we did for the FD method) we define ("cell-centered") control volumes Ω_p around the grid points (see Fig. 1.16),

$$\Omega_p = \left[x_p - \frac{\Delta x}{2}, x + \frac{\Delta x}{2}\right] \times \left[y_p - \frac{\Delta y}{2}, y + \frac{\Delta y}{2}\right]. \tag{1.75}$$

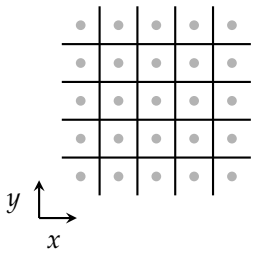


Figure 1.16: Cell-centered control volumes.

Here, we used the regular grid of points defined in Eq. (1.70), but the FV method can also accomodate irregular control volumes. Then, we can integrate Eq. (1.74) over each control volume Ω_p,

$$-\int_{\Omega_p} \partial_i v^i \,\mathrm{d}V = -\int_{\partial\Omega_p} n_i v^i \,\mathrm{d}A = \int_{\Omega_p} f(\boldsymbol{x}) \,\mathrm{d}V\,, \tag{1.76}$$

where we have applied the divergence theorem to integrate the flux over the boundary of the control volume with unit normal n_i instead. In one dimension, Eq. (1.76) reduces to

$$-v_{p+1/2} + v_{p-1/2} \approx \Delta x\, f_p, \tag{1.77}$$

where the flux v is evaluated midway between the grid points, and we have approximated the right-hand side integral with the midpoint rule. Further approximating the flux as $v_{p\pm1/2} \approx \pm\left(u_{p\pm1} - u_p\right)/\Delta x$ reduces Eq. (1.77) to the one-dimensional FD approximation, Eq. (1.73). The focus on conservation laws, which are naturally formulated in terms of fluxes and ubiquitous in physics, makes FV methods popular in many areas of physics, such as computational fluid dynamics.

Evidently, the FD and FV methods are exceptionally straightforward to implement for the simple Poisson problem. They also generalize well to higher dimensions and more involved equations, and are the method of choice in many numerical relativity and computational fluid dynamics codes due to their robustness and relative easy of implementation. However, the numerical error introduced by truncating the Taylor expansion, Eq. (1.72), decreases only polynomially with the grid spacing, with a power determined by the order of the approximation. Therefore, FD and FV methods can quickly exhaust the available computational resources with increasing resolution. Higher-order FD stencils are not particularly well-suited for parallelization since they depend on an order-dependent number of neighboring grid points that must be communicated between processors.

1.3.2 Spectral methods

Spectral methods expand the solution fields in a functional basis and furnish a discretization by truncating the expansion at a suitable order [125].

[125]: Boyd (2001), *Chebyshev and Fourier Spectral Methods*

A field $u(\boldsymbol{x})$ in d dimensions is approximated by the finite series

$$u(\boldsymbol{x}) \approx u_N(\boldsymbol{x}) = \sum_{p=1}^{N} \tilde{u}_p \phi_p(\boldsymbol{x}), \tag{1.78}$$

where $\phi_p(\boldsymbol{x})$ are N basis functions and $\tilde{u}_p$ are the corresponding spectral coefficients, or *modes*.[32] The choice of basis can be informed by the problem at hand. Common choices are Fourier series for periodic problems, spherical harmonics for the angular directions of spherical problems, and Chebyshev or Legendre polynomials for (deformed) cubes (illustrated in Fig. 1.17).[33] For an orthonormal basis with inner product $\left(\phi_p, \phi_q\right) = \delta_{pq}$ the modes of a function $u(\boldsymbol{x})$ can be computed as

$$\tilde{u}_p = \left(u, \phi_p\right). \tag{1.79}$$

For example, Chebyshev polynomials $T_n(x)$ are orthonormal with respect to the inner product

$$\left(T_m, T_n\right) \equiv \frac{2 - \delta_{n0}}{\pi} \int_{-1}^{1} T_m(x) T_n(x) \frac{\mathrm{d}x}{\sqrt{1 - x^2}} = \delta_{mn}, \tag{1.80}$$

and Legendre polynomials $P_n(x)$ are orthonormal with

$$\left(P_m, P_n\right) \equiv \frac{2n + 1}{2} \int_{-1}^{1} P_m(x) P_n(x) \,\mathrm{d}x = \delta_{mn}. \tag{1.81}$$

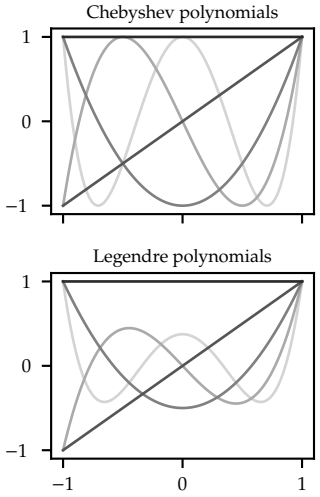


Figure 1.17: The first five Chebyshev and Legendre polynomials.

Derivatives of the discrete field $u_N(\boldsymbol{x})$ can be obtained from the spectral expansion (1.78) and expanded in the basis as well,

$$\partial_i u_N(\boldsymbol{x}) = \sum_{q=1}^{N} \tilde{u}_q \partial_i \phi_q(\boldsymbol{x}) \approx \sum_{q=1}^{N} \tilde{u}_q \sum_{p=1}^{N} \mathrm{D}_{i,pq} \phi_p(\boldsymbol{x}). \tag{1.82}$$

Here, the coefficients of the spectral *differentiation matrix* $\mathrm{D}_{i,pq}$ are

$$\mathrm{D}_{i,pq} = \left(\phi_p, \partial_i \phi_q\right), \tag{1.83}$$

so the modes of the derivative $\partial_i u$ can be computed by matrix multiplication with the modes of the field, $\mathrm{D}_i \underline{\tilde{u}}$. To discretize an elliptic equation, such as the Poisson equation (1.35), we can now insert the truncated expansion (1.82) to find

$$-\sum_{p=1}^{N} \left(\mathrm{D}_i \mathrm{D}_i \underline{\tilde{u}}\right)_p \phi_p(\boldsymbol{x}) = f(\boldsymbol{x}). \tag{1.84}$$

We have to find a way to extract N conditions from this equation to determine the N modes $\underline{\tilde{u}}$ of the solution. A common strategy is the *pseudospectral* method, where we evaluate the equations at N *collocation points* $\boldsymbol{x}_p$. This strategy restores the notion of a computational grid to the spectral method. In fact, the representation of a function $u(\boldsymbol{x})$ in terms of N modes is equivalent to its representation in terms of the values at N collocation points, or *nodes*, $\underline{u}$. From the nodal perspective, the function

32: The number of basis functions is often counted as $N + 1$ in the literature, so N is the highest degree of a polynomial basis. In this thesis, I denote the polynomial degree as P and reserve $N = P + 1$ for the number of basis functions, or the number of grid points.

33: Boyd [125] gives useful advice on the choice of basis functions in his "Moral Principle 1".

(1.35): $-\Delta u = f(\boldsymbol{x})$

is expanded in Lagrange interpolating polynomials,

$$u(\boldsymbol{x}) = \sum_{p=1}^{N} u_p \psi_p(\boldsymbol{x}), \quad \psi_p(\boldsymbol{x}) = \prod_{i=1}^{d} \ell_{p_i}(x^i), \quad \ell_{p_i}(x) = \prod_{\substack{q_i=1 \\ q_i \neq p_i}}^{N_i} \frac{x - x_{q_i}}{x_{p_i} - x_{q_i}}, \tag{1.85}$$

where $\ell_{p_i}(x)$ are the one-dimensional Lagrange polynomials rooted at the N_i collocation points x_{p_i} in direction i of the grid, so $\psi_p(\boldsymbol{x}_q) = \delta_{pq}$ (illustrated in Fig. 1.18). The *Vandermonde matrix* $\mathcal{V}$ transforms between the modal and the nodal representation,

$$\mathcal{V}\underline{\tilde{u}} = \underline{u}, \qquad \mathcal{V}_{pq} = \phi_q(\boldsymbol{x}_p). \tag{1.86}$$

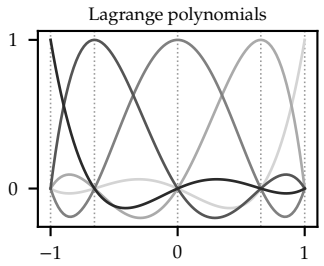


Figure 1.18: Lagrange interpolating polynomials rooted at five Legendre-Gauss-Lobatto (LGL) collocation points (dotted lines).

In principle, we are free to choose any set of collocation points. However, equidistant grid points such as those constructed for finite differencing, Eq. (1.70), behave poorly for the interpolation.[34] Instead, good choices for collocation points are either Gauss quadrature points (the zeros of the highest-order Chebyshev or Legendre polynomial) or Gauss-Lobatto quadrature points (the extrema of the next-to-highest-order polynomial plus points at the boundary at -1 and 1).

34: See the discussion in Hesthaven and Warburton [126], Section 3.1.

[126]: Hesthaven and Warburton (2008), *Nodal Discontinuous Galerkin Methods*

Evaluating Eq. (1.84) at the collocation points, the pseudospectral discretization of the Poisson equation (1.35) reduces to a matrix equation of the form (1.68),

$$\mathcal{A}\underline{\tilde{u}} = \underline{f}, \quad \text{with} \quad \mathcal{A} = -\mathcal{V}\mathrm{D}_i\mathrm{D}_i. \tag{1.87}$$

(1.35): $-\Delta u = f(\boldsymbol{x})$

In contrast to the finite-difference matrix representation of the Poisson equation depicted in Fig. 1.15, the spectral representation is not sparse (except for sparsity introduced by the decoupling of grid points in multiple dimension). Therefore, a spectral matrix can be more computationally intensive to store and to invert than a finite-difference matrix. However, the spectral discretization has the supreme advantage that the numerical error incurred by truncating the expansion Eq. (1.78) decreases *exponentially* with the number of basis functions or grid points, as long as the expanded function is sufficiently smooth [125]. Therefore, matrices representing spectral discretizations can be much smaller than those representing finite-difference discretizations while achieving the same accuracy. While slightly more involved in their implementation, the exponential convergence makes spectral discretizations highly desireable to solve smooth problems at high accuracy, or to conserve computational resources. A spectral discretization is a key component of the discontinuous Galerkin method, outlined in Section 1.3.3 below and developed for applications in numerical relativity in Chapter 2.

[125]: Boyd (2001), *Chebyshev and Fourier Spectral Methods*

1.3.3 Discontinuous Galerkin finite-element methods

Discontinuous Galerkin (DG) methods combine the exponential convergence of a spectral scheme with the mesh-refinement capability of finite element methods. They split the computational domain into nonoverlapping elements and define a spectral discretization in each. The elements are coupled through fluxes across their boundary, borrowing concepts from finite volume methods. Increasing the spectral discretization order within the elements (p refinement) recovers exponential convergence

where the solution is smooth, and adjusting the size and shape of elements (h refinement) can adapt the scheme to the structure of the solution and to the geometry of the computational domain. See Hesthaven and Warburton [126] for a detailed introduction to DG methods.

[126]: Hesthaven and Warburton (2008), *Nodal Discontinuous Galerkin Methods*
(1.35): $-\Delta u = f(\boldsymbol{x})$

For the simple Poisson equation (1.35) a DG scheme is readily formulated by employing the spectral methods outlined in Section 1.3.2 above. Within an element Ω_k, we choose a nodal representation of the field $u(\boldsymbol{x})$ in terms of its values $\underline{u}^{(k)}$ on a regular grid of LGL collocation points following Eq. (1.85). Then, instead of evaluating the Poisson equation directly at the collocation points as in Eq. (1.87), we formulate it in first-order flux form,

$$-\partial_i v^i = f(\boldsymbol{x}), \tag{1.88a}$$

$$\partial_i u = v^i, \tag{1.88b}$$

and project it onto the set of element-local nodal basis functions $\psi_p(\boldsymbol{x})$ (*Galerkin* projection),

$$-\int_{\Omega_k} \psi_p(\boldsymbol{x})\, \partial_i v^i \,\mathrm{d}V = \int_{\Omega_k} \psi_p(\boldsymbol{x})\, f(\boldsymbol{x}) \,\mathrm{d}V\,, \tag{1.89a}$$

$$\int_{\Omega_k} \psi_p(\boldsymbol{x})\, \partial_i u \,\mathrm{d}V = \int_{\Omega_k} \psi_p(\boldsymbol{x})\, v^i \,\mathrm{d}V\,. \tag{1.89b}$$

With a partial integration on the left-hand side and expanding both $v^i(\boldsymbol{x})$ and $f(\boldsymbol{x})$ in the nodal basis, Eq. (1.89a) becomes

$$\underbrace{v_q^i \int_{\Omega_k} \partial_i \psi_p(\boldsymbol{x})\, \psi_q(\boldsymbol{x}) \,\mathrm{d}V}_{=:\,\mathrm{MD}_{i,pq}^T} - n_i v_q^i \underbrace{\int_{\partial\Omega_k} \psi_p(\boldsymbol{x})\, \psi_q(\boldsymbol{x}) \,\mathrm{d}A}_{=:\,\mathrm{ML}_{pq}}$$

$$= f_q \underbrace{\int_{\Omega_k} \psi_p(\boldsymbol{x})\, \psi_q(\boldsymbol{x}) \,\mathrm{d}V}_{=:\,\mathrm{M}_{pq}}\,, \tag{1.90}$$

where we have defined the *mass matrix* M, the *stiffness matrix* MD_i, and the *lifting operator* ML on the element. In matrix notation, the discretized Poisson equation on an element Ω_k, and similarly the first-order auxiliary equation (1.88b), are

$$\mathrm{MD}_i^T \cdot \underline{v}^i - \mathrm{ML} \cdot \left(n_i \underline{v}^i\right)^* = \mathrm{M} \cdot \underline{f}, \tag{1.91a}$$

$$-\mathrm{MD}_i^T \cdot \underline{u} + \mathrm{ML} \cdot \left(n_i \underline{u}\right)^* = \mathrm{M} \cdot \underline{v}^i, \tag{1.91b}$$

where the symbol $\cdot$ emphasises matrix multiplication. Here, the quantities $n_i \underline{v}^i$ and $n_i \underline{u}$ are the fluxes normal to the element boundary. They are evaluated only on the element boundary since ML_{pq} is zero for indices p and q corresponding to grid points away from the boundary (see Fig. 1.18). To couple grid points across elements, we promote these boundary fluxes to *numerical fluxes*, denoted by the superscipt *. Whereas we have worked with quantities local to each element thus far (but omitted the superscript $^{(k)}$ on every symbol), the numerical fluxes are functions of grid points on both sides of a shared element boundary. The discontinuous Galerkin method is characterized by treating the grid points on either side of the shared boundary as distinct degrees of

freedom, coupled through a numerical flux. For the scheme to be well-posed, the numerical fluxes $\left(n_i v^i\right)^*(n_i, u^{\text{int}}, u^{\text{ext}})$ and $\left(n_i u\right)^*(n_i, u^{\text{int}}, u^{\text{ext}})$ must be *consistent*,

$$\left(n_i v^i\right)^*(n_i, u, u) = n_i v^i, \tag{1.92}$$

and *conservative*,

$$\left(n_i v^i\right)^*(n_i, u^{\text{int}}, u^{\text{ext}}) = -\left(n_i v^i\right)^*(-n_i, u^{\text{ext}}, u^{\text{int}}), \tag{1.93}$$

where "int" denotes the quantities on the *interior* side of the element, "ext" denotes the *exterior* side, and we are assuming $n_i \equiv n_i^{\text{int}} = -n_i^{\text{ext}}$. A commonly employed numerical flux for the Poisson equation is the *symmetric internal penalty* (SIP) flux,

$$\begin{aligned}\left(n_i u\right)^* &= \frac{1}{2} n_i \left(u^{\text{int}} + u^{\text{ext}}\right) \\ &\equiv n_i \left\{\!\!\left\{u\right\}\!\!\right\}, \qquad (1.94a)\\ \left(n_i v^i\right)^* &= \frac{1}{2} n_i \left(\partial_i u^{\text{int}} + \partial_i u^{\text{ext}}\right) - \sigma \left(u^{\text{int}} - u^{\text{ext}}\right) \\ &\equiv n_i \left\{\!\!\left\{\partial_i u\right\}\!\!\right\} - \sigma \left[\!\left[u\right]\!\right]. \qquad (1.94b)\end{aligned}$$

Here, $\sigma \geq C\,(P+1)^2/h$ is the *penalty* with the polynomial degree P orthogonal to the boundary, the size h of the element, and $C \geq 1$. The penalty breaks the degeneracy of the numerical flux, which would admit a solution with $u^{\text{ext}} = -u^{\text{int}}$ for $\sigma = 0$.[35] I have included the double-bracket notation in this presentation that is often found in the DG literature, but I will not use it any further in this thesis to avoid obscuring math with notation.

35: See discussion in Hesthaven and Warburton [126], Section 7.2.

The DG-discretized first-order formulation of the Poisson equation, Eq. (1.91), is of the matrix form (1.68), $\mathcal{A}\underline{w} = \underline{b}$, where both $\underline{w} = \{\underline{u}, \underline{v}^i\}$ and $\underline{b}$ now collect the values at all grid points of all elements Ω_k. It also admits a "primal formulation" that eliminates the auxiliary degrees of freedom $\underline{v}^i$, so it has the form $\mathcal{A}\underline{u} = \underline{b}$.[36] Either way, it is amenable to the strategies for the solution of sparse matrix equations that we will outline next in Section 1.3.4.

36: See Hesthaven and Warburton [126], Section 7.2.2.

A key advantage of DG methods is their locality: the matrix-vector product $\mathcal{A}\underline{u}$ decomposes into an operation local to each element [Eq. (1.91)]. It depends on neighboring elements only through the numerical fluxes. This allows to distribute evaluation of the matrix-vector product to multiple compute cores, communicating only the numerical fluxes between them. In particular, elements remain coupled only to their nearest neighbors even when the order of the spectral discretization within an element increases. In contrast, FD and FV methods require the communication of data in a region that grows with the order of the stencil (for flux reconstruction in the case of FV methods). However, the locality of the DG method comes at the cost of duplicate grid points on element boundaries. For low-order DG elements these duplicate points can constitute a considerable fraction of the total number of grid points. Increasing the order of the DG elements both reduces the fraction of duplicate points and leverages the exponential convergence of the spectral discretization, making the DG method more effective.

The advantages of DG methods—their exponential convergence for

smooth problems, their ability to conform to the geometry of the problem through a choice of elements, and their decomposition into element-local operations—also come at the cost of considerable complexity in implementing the method. Even for the simple Poisson equation presented above we have not yet discussed boundary conditions, curved meshes, mesh refinement, evaluating the mass, stiffness, and lifting operators on an element, and eliminating the auxiliary degrees of freedom. For more involved equations we also have to handle nonlinear terms and boundary conditions, curved manifolds, and suitable choices for the numerical fluxes. My publication [1] presented in Chapter 2 develops a DG scheme with all the bells and whistles needed to construct discrete representations for the nonlinear elliptic equations that we encounter in numerical relativity. It discusses all the details that we have carelessly ignored in this brief sketch of the DG-discretized Poisson equation, and defines a DG scheme that is general enough to accommodate a large class of elliptic equations without having to revisit their DG discretization every time a new set of equations is implemented.

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

1.3.4 Numeric methods for sparse matrix equations

All discretization methods outlined above reduce linear elliptic problems to a matrix equation $\mathcal{A}\underline{u} = \underline{b}$, where the matrix $\mathcal{A}$ couples every degree of freedom in the computational domain to every other. The matrix is typically *sparse*, such as the band-diagonal FD representation of the Poisson equation in Fig. 1.15 or a DG discretization that couples only nearest-neighbor elements. Therefore, numeric solutions to elliptic PDEs fall in the realm of computational methods for sparse matrix equations, which is a rich and evolving area of research. See Saad [127] for a detailed introduction.

[127]: Saad (2003), *Iterative Methods for Sparse Linear Systems*

The most straightforward methods to solve sparse matrix equations involve constructing the matrix explicitly and inverting it directly [128]. These methods take advantage of the sparsity of the matrix by storing only its nonzero entries, and by reducing arithmetic operations to the nonzero entries as well. In cases where the matrix is nontrivial to construct explicitly it is always possible to construct it column-by-column from the matrix-vector product $\mathcal{A}\underline{u}$ as

[128]: Davis (2006), *Direct Methods for Sparse Linear Systems*

$$\mathcal{A}_{pq} = \left(\mathcal{A}\,\underline{e}_q\right)_p, \tag{1.95}$$

Here, $\underline{e}_q = (0, \ldots, 1, \ldots, 0)$ is the qth unit vector that fills the entire computational grid with zeros, except for a one at position q. The construction from unit vectors is very parallelizable and the resulting matrix can also be partitioned and stored on multiple cores. The inversion of the matrix is most commonly accomplished by variations of the *LU decomposition*, where the matrix $\mathcal{A}$ is decomposed into a lower triangular matrix L and an upper triangular matrix U,

$$\mathcal{A} = L\,U. \tag{1.96}$$

Once this decomposition has been computed, linear problems $\mathcal{A}\underline{u} = \underline{b}$ are solved by two sparse triangular matrix inversions, which reduce to one forward and backward sweep over all degrees of freedom. Prominent software packages for the direct solution of sparse matrix equations include

`SuperLU` [129], `MUMPS` [130], and `UMFPACK` [131]. For example, Python's `scipy.linalg.sparse` module (version 1.8.0) [132] uses `SuperLU` for sparse LU decompositions.

[129]: Li (2005), *An Overview of SuperLU: Algorithms, Implementation, and User Interface*

[130]: Amestoy et al. (2001), *A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling*

[131]: Davis (2004), *Algorithm 832: UMFPACK V4.3—an Unsymmetric-Pattern Multifrontal Method*

[132]: Virtanen et al. (2020), *SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python*

Direct solution methods are feasible for one- or two-dimensional elliptic problems, for three-dimensional problems at modest resolution, or perhaps still for higher-resolution three-dimensional problems with a single or few variables per grid point, such as Poisson-type problems. Some recent developments in numerical relativity resort to direct methods for the solution of elliptic problems. Rashti et al. [108] develop a technique to split the matrix equation into smaller subproblems so it is amenable to direct solution methods, and they solve the coupled elliptic XCTS equations (Section 1.2.3) sequentially in a fixed-point iteration procedure to reduce the size of the matrices. Papenfort et al. [89] employ the `KADATH` library [133], which constructs the matrix explicitly column-by-column, distributes it to a large number of cores, and solves it using the `ScaLAPACK` library [134]. They report that the explicit matrix construction and its solution take a comparative amount of time, that the code scales to as many as 32 k cores, and that it requires a large amount of memory. They suggest iterative methods with suitable preconditioners to reduce the excessive amount of computational resources needed to solve elliptic problems. Such methods are the subject of this thesis and will be discussed next.

[108]: Rashti et al. (2021), *Elliptica: a new pseudo-spectral code for the construction of initial data*

[89]: Papenfort et al. (2021), *New public code for initial data of unequal-mass, spinning compact-object binaries*

[133]: Grandclément (2010), *KADATH: A spectral solver for theoretical physics*

[134]: Blackford et al. (1997), *ScaLAPACK Users' Guide*

For coupled, three-dimensional elliptic equations at high resolutions, when the matrix $\mathcal{A}$ couples millions of degrees of freedom or more, direct solution methods are not feasible anymore. Instead, iterative algorithms are the method of choice [127]. Iterative methods find the solution $\underline{u}$ to a matrix equation $\mathcal{A}\underline{u} = \underline{b}$ at incrementally increasing accuracy in a series of steps, beginning at an initial guess $\underline{u}_0$ for the solution. They do not require the matrix representation $\mathcal{A}$ explicitly but rely only on the matrix-vector product $\mathcal{A}\underline{u}$. Basic relaxation algorithms such as the Jacobi, Gauss-Seidel, Successive Overrelaxation (SOR), and Richardson methods fall into this category. They are all defined by the simple fixed-point iteration procedure

[127]: Saad (2003), *Iterative Methods for Sparse Linear Systems*

$$\underline{u}_{k+1} = \underline{u}_k + \mathcal{M}^{-1}\left(\underline{b} - \mathcal{A}\underline{u}_k\right), \tag{1.97}$$

where the index k enumerates the iterations, and the matrix $\mathcal{M}$ depends on the specific method. For instance, the Jacobi method is defined simply by $\mathcal{M} = \operatorname{diag}\mathcal{A}$, the diagonal entries of $\mathcal{A}$. Generally, a suitable choice for the matrix $\mathcal{M}$ is an easily-invertible approximation of $\mathcal{A}$. It is referred to as a *preconditioner*. From this perspective, relaxation methods are equivalent to fixed-point iterations on the preconditioned system,[37]

37: See Saad [127], Section 4.1.2.

$$\mathcal{M}^{-1}\mathcal{A}\underline{u} = \mathcal{M}^{-1}\underline{b}. \tag{1.98}$$

The relaxation methods converge too slowly to be of great use by themselves, but their simplicity and low computational cost per iteration makes them useful to support more advanced methods.

The primary class of iterative algorithms for the solution of sparse matrix equations is the family of Krylov-subspace methods, specifically variations of the conjugate gradients (CG) and generalized minimum residual (GMRES) algorithms [135].[38] The CG algorithm is restricted to symmetric positive definite matrices, whereas GMRES supports general

[135]: Saad and Schultz (1986), *GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems*

38: See Chapter 6 in Saad [127].

matrices at slightly higher computational cost. Both are based on the idea to build up a basis of the Krylov subspace

$$\mathcal{K}_k(\mathcal{A}, \underline{b}) = \text{span}\left\{\underline{b}, \mathcal{A}\underline{b}, \mathcal{A}^2\underline{b}, \ldots, \mathcal{A}^{k-1}\underline{b}\right\} \tag{1.99}$$

so they can approximate the solution $\underline{u}$ by a polynomial in $\mathcal{A}$. The CG method constructs an $\mathcal{A}$-orthogonal basis, where basis vectors $\underline{p}_k$ and $\underline{p}_{k+1}$ are orthogonal with respect to the Euclidean inner product in the conjugate sense, $\left(\underline{p}_k, \mathcal{A}\underline{p}_{k+1}\right) = 0$. The GMRES method constructs the basis using an Arnoldi orthogonalization procedure so a new basis vector is orthogonal to all existing ones, and solves a least-squares problem for the solution vector.

Krylov-subspace methods are guaranteed to converge in at most N_{DOF} iterations for a matrix $\mathcal{A}$ of size $N_{\text{DOF}} \times N_{\text{DOF}}$, since at that point the constructed basis is complete. However, for typical applications the number N_{DOF} is prohibitively large to provide a reasonable upper bound for the number of iterations, so the relevant metric for the quality of the algorithm is its convergence rate. Krylov-subspace methods do not generally converge fast enough by themselves, but require effective *preconditioning*.[39] Given a preconditioner $\mathcal{M}$, such as any of the relaxation methods mentioned above, the Krylov-subspace method solves the left-preconditioned problem (1.98), or the right-preconditioned problem

$$\mathcal{A}\mathcal{M}^{-1}\underline{x} = \underline{b}, \qquad \underline{u} = \mathcal{M}^{-1}\underline{x}. \tag{1.100}$$

For a good preconditioner, this modified problem has better convergence properties than the unpreconditioned problem.

In practice, the preconditioner supports every iteration of the Krylov-subspace method with an approximate solution to accelerate convergence. In a turn of perspective, the preconditioner is often the centerpiece of the iterative algorithm and the outer Krylov solver is considered an accelerator. The challenge is to conceive preconditioners that utilize computational resources effectively, and that complement the weaknesses of the Krylov-subspace algorithm. Preconditioners are also often nested, complementing each other, and can take the problem-specific structure of the discretization into account that led to the linear operator $\mathcal{A}$. I develop a stack of nested preconditioners for the solution of DG-discretized elliptic PDEs in my publication [2] presented in Chapter 3, involving multigrid methods, a highly-parallelizable additive Schwarz smoother on every grid, and optimized kernels for the solution of subproblems based on approximating the PDEs as uncoupled Poisson equations. I place particular focus on the parallelization properties of the computational methods.

Prominent general software packages for the iterative solution of sparse matrix equations include `PETSc` [136], `Trilinos` [137], and `hypre` [138]. For example, the `SpEC` elliptic solver [139] dispatches to `PETSc`, and the `TwoPunctures` code [88] to `hypre`. In this thesis, I base my implementation on the task-based parallization and domain decomposition infrastructure provided by the `SpECTRE` code, which I will discuss in the following sections.

39: See Chapter 9 in Saad [127].

(1.98): $\mathcal{M}^{-1}\mathcal{A}\underline{u} = \mathcal{M}^{-1}\underline{b}$

[2]: **Vu** et al. (2022), *A scalable elliptic solver with task-based parallelism for the `SpECTRE` code*. Chapter 3 of this thesis.

[136]: PETSc, `https://www.mcs.anl.gov/petsc`

[137]: Trilinos (2022), *The Trilinos Project*

[138]: Falgout, Jones, and Yang (2006), *The Design and Implementation of hypre, a Library of Parallel High Performance Preconditioners*

[139]: Pfeiffer et al. (2003), *A multidomain spectral method for solving elliptic equations*

[88]: Ansorg, Brügmann, and Tichy (2004), *A Single-domain spectral method for black hole puncture data*

1.3.5 Task-based parallelism

Parallelization is not an afterthought anymore when developing computational methods in this day and age. Every contemporary consumer laptop has at least four compute cores, and high-performance computing (HPC) clusters with hundreds of thousands interconnected cores are readily available, with even larger clusters in development. Leveraging these vast computational resources for numerical simulations is essential in the quest to simulate astrophysical scenarios in three dimensions, with detailed microphysics, at high resolution, and on realistic timescales (see also the discussion in Section 1.1.6).

The industry standard for the parallelization of grid-based numerical codes is the message passing interface (MPI). Typically, the computational grid is partitioned into regions with similar numbers of points that are distributed to the available cores. Before performing operations that require data residing on different cores, such as an application of the discrete operator matrix $\mathcal{A}$ across the entire grid, the program communicates the data between cores. This is the data-centric approach to parallelism adopted by the majority of simulation codes. It is comparatively straightforward to implement by alternating between global communication and computation phases. However, the number of cores that such globally-synchronous MPI-parallelized programs can effectively scale to is limited. In particular, it becomes challenging to overlap communication and computation, risking that some cores remain idle while others are still performing computations.

Task-based parallel programs approach parallelization by distributing units of work ("tasks") among cores, instead of partitioning the data. The tasks depend on each other by inputs and outputs, and a runtime system schedules the tasks on the available cores. Data associated with each task is migrated between cores alongside the task. Task-based parallel programs can reduce the time that cores spend idle, assuming the algorithmic dependencies allow it. However, implementing a runtime system that controls the execution of tasks and migrates them between cores is a nontrivial challenge of computer science. Therefore, a number of software libraries are being developed to provide the foundation for task-based parallel programs. See Thoman et al. [140] for a recent overview focused on HPC applications.

[140]: Thoman et al. (2018), *A taxonomy of task-based parallel programming technologies for high-performance computing*

My new elliptic solver for numerical relativity, and the `SpECTRE` code as a whole, are based on the `Charm++` task-based parallelism library [141, 142]. The `Charm++` runtime system is driven by *chares* exchanging *messages*. A chare is an object that can migrate between cores and defines *entry methods*. Chares can invoke entry methods on *proxies* of each other irrespective of the core they reside on. The runtime system resolves the proxy and invokes the entry method by delivering a message to the target chare, dispatching a task. Messages can include data needed by the task. `Charm++` can utilize a number of communication backends for the exchange of data between cores, including MPI and low-level machine networking layers such as Verbs or UCX. See Kidder et al. [142] for details on our use of `Charm++`.

[141]: The Charm++ Parallel Programming System, `https://charm.cs.illinois.edu`
[142]: Kidder et al. (2017), *SpECTRE: A Task-based Discontinuous Galerkin Code for Relativistic Astrophysics*

Algorithms in computational physics are generally not yet designed with task-based parallelism in mind. In Chapter 3 I study the parallelization

structure of iterative methods for the solution of sparse matrix equations (as introduced in Section 1.3.4) and develop algorithms that work well in a task-based parallel environment. In particular, discontinuous Galerkin methods are well-suited for task-based parallelism because they decompose naturally into elements that can be mapped to chares. Communication between the elements can be accomplished by messages, and spectral operations on the elements represent units of work performed by tasks.

1.3.6 The SpECTRE numerical relativity code

[10]: SpECTRE, spectre-code.org

The new `SpECTRE` code [10] provides the foundation for my elliptic solver. `SpECTRE` is a collaborative research project primarily targeted at multi-messenger astrophysical sources of gravitational waves such as binary neutron star mergers, and also serves as a platform for interdisciplinary problems involving hyperbolic and elliptic PDEs. It is being developed by the Simulating eXtreme Spacetimes (SXS) collaboration to succeed the Spectral Einstein Code (`SpEC`) [55]. With `SpECTRE` we intend to overcome `SpEC`'s limited parallelizability while retaining its strength in the use of higher-order (spectral) discretization methods.

[55]: Spectral Einstein Code (SpEC), `black-holes.org/code/SpEC`

The `SpECTRE` code is being developed open-source to encourage community engagement and to facilitate reproducibility of scientific results. It is primarily written in `C++17` with a focus on template metaprogramming. Tooling and bindings are written in Python, and input-file configurations to run executables with user-defined parameters are written in YAML (see Appendix B for examples). Executables can be compiled with recent GCC and Clang compilers on Linux and macOS machines, including a selection of supported HPC clusters and a containerized development environment. A strong focus on unit testing intends to maintain the correctness of the code and of scientific output. A continuous integration (CI) and continuous deployment (CD) infrastructure automates the testing procedure and furnishes monthly releases, which are published to GitHub and Zenodo. Contributions to the code adopt a code review workflow to maintain the quality of the code, spread knowledge of the code base among the community, and to strive for good documentation. I serve as a core developer of the `SpECTRE` project since 2020, contributing to core infrastructure of the code, taking on some project-management responsibilities, initiating and signing off on code reviews, and overseeing the code as a whole. At the moment, approximately 10 to 15 developers contribute regularly to the code.

Discontinuous Galerkin scheme for elliptic equations

2

Publication

This chapter is based on the article *Unified discontinuous Galerkin scheme for a large class of elliptic equations* [1], published in *Phys. Rev. D* ***105****, 024034* on Jan 11, 2022 (arXiv:2108.05826). It develops the discontinuous Galerkin discretization scheme for elliptic equations that underpins my new elliptic solver. It also stands on its own as a versatile numerical discretization scheme for elliptic problems in computational physics.

Authors **Nils L. Fischer** and Harald P. Pfeiffer

Abstract We present a discontinuous Galerkin internal-penalty scheme that is applicable to a large class of linear and nonlinear elliptic partial differential equations. The unified scheme can accommodate all second-order elliptic equations that can be formulated in first-order flux form, encompassing problems in linear elasticity, general relativity, and hydrodynamics, including problems formulated on a curved manifold. It allows for a wide range of linear and nonlinear boundary conditions, and accommodates curved and nonconforming meshes. Our generalized internal-penalty numerical flux and our Schur-complement strategy of eliminating auxiliary degrees of freedom make the scheme compact without requiring equation-specific modifications. We demonstrate the accuracy of the scheme for a suite of numerical test problems. The scheme is implemented in the open-source `SpECTRE` numerical relativity code.

Declaration of authorship I am the lead author who wrote this article, argued the direction it should take, developed the numerical scheme presented in the article, implemented the scheme in the `SpECTRE` code, and performed the numerical computations applying the scheme to test problems. Harald Pfeiffer acted as advisor in this project and provided editorial feedback on the article. Aside from this, the work is my own.

2.1 Introduction

Many problems in physics involve the numerical solution of second-order elliptic partial differential equations (PDEs). Such elliptic problems often represent static field configurations under the effect of external forces and arise, for example, in electrodynamics, in linear or nonlinear elasticity, and in general relativity. Elliptic problems also often accompany time evolutions, where they constrain the evolved fields at every instant in time or provide admissible initial data for the evolution.

Discontinuous Galerkin (DG) methods are gaining popularity in the computational physics and engineering community and are currently most prevalently used for time evolutions of hyperbolic boundary-value

problems [126, 143–145]. Many properties that make DG methods advantageous for time evolutions also apply to elliptic problems, which lead to the development of DG schemes for elliptic PDEs [146, 147]. In particular, DG schemes provide a flexible mechanism for refining the computational grid, retaining exponential convergence even in the presence of discontinuities when adaptive mesh refinement (AMR) techniques are employed [8, 148]. Furthermore, some difficulties with DG schemes in time evolutions, such as shock capturing, are not present in elliptic problems and their static nature makes it often (but not always) straightforward to place grid boundaries at discontinuities, thus relieving the AMR scheme from the responsibility of resolving them. See, e.g., Ref. [126] and the seminal paper [147] for extensive discussions of DG schemes for the Poisson equation, and Refs. [149–151] for discussions of linear and nonlinear elasticity.

In the context of relativistic astrophysics and numerical relativity, DG methods have been developed for hyperbolic equations on curved manifolds thus far [142, 152, 153]. In Ref. [8] we explored the feasibility of the DG method for elliptic problems in numerical relativity confined to flat Poisson-type equations with nonlinear sources. In this article we present a DG scheme suitable to solve a significantly larger class of elliptic problems that arise in numerical relativity. Most notably, the scheme encompasses the extended conformal thin sandwich (XCTS) formulation of the general-relativistic Einstein constraint equations on a curved manifold, and associated boundary conditions [74, 113]. Solutions to the XCTS equations provide admissible initial data for general-relativistic time evolutions, for scenarios such as two orbiting black holes or neutron stars [75, 78, 109, 112]. To our knowledge, this article presents the first discontinuous Galerkin solution of the full Einstein constraint equations. Aimed at applications in numerical relativity, the scheme is implemented in the publicly available `SpECTRE` code [2, 10].

Furthermore, the elliptic DG scheme presented in this article is not limited to applications in numerical relativity. It applies to all second-order elliptic problems that can be formulated in first-order flux form. Besides the classic Poisson and elasticity equations it covers a large class of elliptic problems in general relativity and hydrodynamics, including coupled systems of equations and those formulated on a curved manifold. With our unified DG scheme, new elliptic systems can be implemented by supplying their first-order fluxes and sources, hence no knowledge of the DG technology or of finite-element formulations is required. This lowers the barrier for extending the capabilities of a simulation code. We pay particular attention to support a wide range of linear and nonlinear boundary conditions so our DG scheme is suited to solve many real-world scenarios (as well as some out-of-this-world scenarios such as initial data for evolutions of black holes and neutron stars). We are aware only of Ref. [154] studying a nonlinear boundary condition for an elliptic DG problem.

To formulate the unified DG scheme we present a generalized internal-penalty numerical flux, which avoids problem-specific parameters that are needed, e.g., in Refs. [155–157]. We eliminate auxiliary degrees of freedom that arise from the first-order form with a Schur-complement strategy, which has proven more suitable to the unified DG scheme than primal formulations that are commonly employed in the literature [126,

[126]: Hesthaven and Warburton (2008), *Nodal Discontinuous Galerkin Methods*
[143]: Reed and Hill (1973), *Triangular mesh methods for the neutron transport equation*
[144]: Cockburn, Karniadakis, and Shu (2000), *The Development of Discontinuous Galerkin Methods*
[145]: Cockburn (2001), *Devising discontinuous Galerkin methods for non-linear hyperbolic conservation laws*

[146]: Wheeler (1978), *An Elliptic Collocation-Finite Element Method with Interior Penalties*
[147]: Arnold et al. (2002), *Unified Analysis of Discontinuous Galerkin Methods for Elliptic Problems*

[8]: Vincent, Pfeiffer, and **Fischer** (2019), *hp-adaptive discontinuous Galerkin solver for elliptic equations in numerical relativity*
[148]: Schötzau et al. (2014), *Exponential Convergence of hp-DGFEM for Elliptic Problems in Polyhedral Domains*

[149]: Hansbo and Larson (2002), *Discontinuous Galerkin methods for incompressible and nearly incompressible elasticity by Nitsche's method*
[150]: Cockburn, Schötzau, and Wang (2006), *Discontinuous Galerkin methods for incompressible elastic materials*
[151]: Ortner and Süli (2007), *Discontinuous Galerkin Finite Element Approximation of Nonlinear Second-Order Elliptic and Hyperbolic Systems*

[142]: Kidder et al. (2017), *SpECTRE: A Task-based Discontinuous Galerkin Code for Relativistic Astrophysics*
[152]: Teukolsky (2016), *Formulation of discontinuous Galerkin methods for relativistic astrophysics*
[153]: Fambri et al. (2018), *ADER discontinuous Galerkin schemes for general-relativistic ideal magnetohydrodynamics*

[2]: **Vu** et al. (2022), *A scalable elliptic solver with task-based parallelism for the `SpECTRE` code*. Chapter 3 of this thesis.
[10]: SpECTRE, spectre-code.org

[154]: Feistauer, Roskovec, and Sändig (2019), *Discontinuous Galerkin method for an elliptic problem with nonlinear Newton boundary conditions in a polygon*

[155]: Hartmann and Houston (2008), *An optimal order interior penalty discontinuous Galerkin discretization of the compressible Navier–Stokes equations*
[156]: Epshteyn and Rivière (2007), *Estimation of penalty parameters for symmetric interior penalty Galerkin methods*
[157]: De Basabe, Sen, and Wheeler (2008), *The interior penalty discontinuous Galerkin method for elastic wave propagation: grid dispersion*

147]. The resulting DG scheme is compact, in the sense that it involves only nearest-neighbor couplings and no auxiliary degrees of freedom, and symmetric, unless the symmetry is broken by the elliptic equations.

This article is structured as follows. Section 2.2 details the generic first-order flux formulation that serves as the starting point for our DG discretization. Section 2.3 develops the unified DG scheme. In Section 2.4 we apply the DG scheme to a set of increasingly challenging test problems. The test problems include scenarios derived from general relativity that feature sets of coupled, strongly nonlinear equations on a curved manifold with nonlinear boundary conditions, solved on curved meshes. We conclude in Section 2.5.

2.2 First-order flux formulation

We consider second-order elliptic PDEs of one or more "primal" variables $u_A(\boldsymbol{x})$, where the index A labels the variables. The variables can be scalars (like in the Poisson equation) or tensorial quantities (like in an elasticity problem). We reduce the PDEs to first order by introducing "auxiliary" variables $v_A(\boldsymbol{x})$, which typically are gradients of the primal variables. We then restrict our attention to problems that can be formulated in first-order flux form

$$-\partial_i \mathcal{F}_\alpha{}^i[u_A, v_A; \boldsymbol{x}] + \mathcal{S}_\alpha[u_A, v_A; \boldsymbol{x}] = f_\alpha(\boldsymbol{x}), \tag{2.1}$$

where the index α enumerates both u_A and v_A. Here, the fluxes $\mathcal{F}_\alpha{}^i$ and the sources $\mathcal{S}_\alpha$ are functionals of the variables u_A and v_A, but not their derivatives, as well as the coordinates $\boldsymbol{x}$. The fixed sources $f_\alpha(\boldsymbol{x})$ are independent of the variables. Lowercase Latin indices i, j, k, l enumerate spatial dimensions, and we employ the Einstein sum convention to sum over repeated indices.

The flux form (2.1) is general enough to encompass a wide range of elliptic problems. For example, a flat-space Poisson equation in Cartesian coordinates,

$$-\partial_i \partial_i u(\boldsymbol{x}) = f(\boldsymbol{x}), \tag{2.2}$$

has the single primal variable $u(\boldsymbol{x})$. Choosing the auxiliary variable $v_i = \partial_i u$ we can formulate the Poisson equation with the fluxes and sources

$$\mathcal{F}_{v\,j}{}^i = u\,\delta^i_j, \quad \mathcal{S}_{v\,j} = v_j, \quad f_{v\,j} = 0, \tag{2.3a}$$

$$\mathcal{F}_u{}^i = v_i, \qquad \mathcal{S}_u = 0, \qquad f_u = f(\boldsymbol{x}), \tag{2.3b}$$

where δ^i_j denotes the Kronecker delta. Note that Eq. (2.3a) is the definition of the auxiliary variable, and Eq. (2.3b) is the Poisson equation (2.2).

The equation of linear elasticity in Cartesian coordinates,

$$-\partial_i Y^{ijkl} \partial_{(k}\, u_{l)} = f^j(\boldsymbol{x}), \tag{2.4}$$

has the primal variable $u^i(\boldsymbol{x})$, describing the vectorial deformation of an elastic material. The constitutive relation $Y^{ijkl}(\boldsymbol{x})$ captures the elastic properties of the material in the linear regime. Choosing the symmetric

strain $S_{ij} = \partial_{(i} u_{j)} = (\partial_i u_j + \partial_j u_i)/2$ as auxiliary variable we can formulate the elasticity equation with the fluxes and sources

$$\mathcal{F}_S{}^i_{jk} = \delta^i_{(j} u_{k)}, \quad \mathcal{S}_{S\,jk} = S_{jk}, \quad f_{S\,jk} = 0, \tag{2.5a}$$

$$\mathcal{F}_u{}^{ij} = Y^{ijkl} S_{kl}, \quad \mathcal{S}_u{}^j = 0, \quad f_u{}^j = f^j(\boldsymbol{x}). \tag{2.5b}$$

Again, Eq. (2.5a) is the definition of the auxiliary variable and Eq. (2.5b) is the elasticity equation (2.4). The fluxes and sources for the elasticity system (2.5) have higher rank than those for the Poisson system (2.3).

The first-order flux form (2.1) also accommodates equations formulated on a curved manifold which is equipped with a metric $g_{ij}(\boldsymbol{x})$. Such equations typically involve covariant derivatives ∇_i compatible with g_{ij}. To formulate the equations in flux form (2.1) we expand covariant derivatives in partial derivatives and Christoffel symbols $\Gamma^i_{jk} = \frac{1}{2} g^{il} \left(\partial_j g_{kl} + \partial_k g_{jk} - \partial_l g_{jk}\right)$. Christoffel symbols also appear when formulating equations in curvilinear coordinates. In our scheme, the terms with partial derivatives are assigned to the fluxes $\mathcal{F}^i$ and the terms with Christoffel symbols are assigned to the sources $\mathcal{S}$. For example, a curved-space Poisson equation

$$-g^{ij} \nabla_i \nabla_j u(\boldsymbol{x}) = f(\boldsymbol{x}) \tag{2.6}$$

with auxiliary variable $v_i = \nabla_i u$ can be formulated with the fluxes and sources

$$\mathcal{F}_v{}^i_j = u\, \delta^i_j, \quad \mathcal{S}_{v\,j} = v_j, \quad f_{v\,j} = 0, \tag{2.7a}$$

$$\mathcal{F}_u{}^i = g^{ij} v_j, \quad \mathcal{S}_u = -\Gamma^i_{ij} g^{jk} v_k, \quad f_u = f(\boldsymbol{x}). \tag{2.7b}$$

Our strategy of expanding covariant derivatives differs from the formulations employed for relativistic hyperbolic conservation laws by Teukolsky [152], where fluxes are always vector fields and therefore the covariant divergence can always be written in terms of partial derivatives and the metric determinant.[1] In contrast, fluxes in the elliptic equations (2.1) can be higher-rank tensor fields, as exemplified in Eq. (2.5).

[152]: Teukolsky (2016), *Formulation of discontinuous Galerkin methods for relativistic astrophysics*

1: See Eq. (2.3) in Ref. [152].

(2.1): $-\partial_i \mathcal{F}_\alpha{}^i[u_A, v_A; \boldsymbol{x}] + \mathcal{S}_\alpha[u_A, v_A; \boldsymbol{x}] = f_\alpha(\boldsymbol{x})$

The fixed sources $f_\alpha(\boldsymbol{x})$ could, in principle, be absorbed in the sources $\mathcal{S}_\alpha$. However, it is useful to keep these variable-independent contributions separate for two reasons. First, they remain constant throughout an elliptic solve, so they need not be recomputed when the dynamic variables change. Second, the constant contributions represent a nonlinearity in the variables u_A and v_A when included in the sources $\mathcal{S}_\alpha$. Assigning the constant contributions to the fixed sources f_α eliminates this particular nonlinearity, hence allowing us to avoid an explicit linearization procedure if the remaining sources $\mathcal{S}_\alpha$ are linear.

The Appendix in Ref. [1] lists fluxes and sources for selected elliptic problems.[2] Our focus on systems in generic first-order flux form allows us to solve a variety of elliptic systems by only implementing their fluxes and sources. We now proceed to discretize this generic formulation.

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

2: Appendix omitted in this thesis because the introductory Section 1.2 includes the relevant equations and additional context.

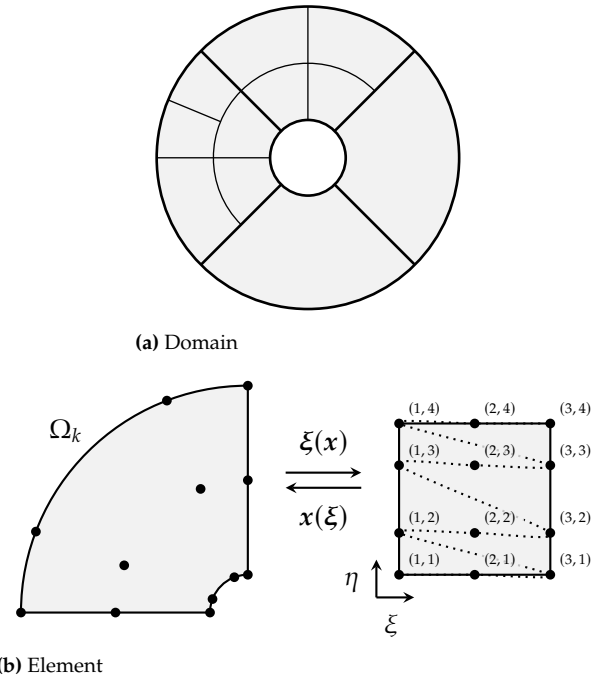


Figure 2.1: *Top:* Geometry of a two-dimensional computational domain composed of four wedge-shaped blocks. Each block is split in one or more nonoverlapping elements Ω_k. *Bottom:* The coordinate transformation $\boldsymbol{\xi}(\boldsymbol{x})$ maps an element to a reference cube $[-1,1]^2$ with logical coordinate axes $\boldsymbol{\xi} = (\xi, \eta)$. In this example we chose $N_{k,\xi} = 3$ and $N_{k,\eta} = 4$ Legendre-Gauss-Lobatto collocation points along ξ and η, respectively. Each grid point is labeled with its index (p_ξ, p_η). The dotted line connects points in the order they are enumerated in by the index p.

2.3 DG discretization of the flux formulation

In this section we develop the unified DG scheme for elliptic equations in flux form, Eq. (2.1). Novel features of our scheme are the formulation of DG residuals and boundary conditions in terms of generic fluxes and sources of arbitrary tensor rank (Sections 2.3.2 and 2.3.5), and the generalized internal-penalty numerical flux (Section 2.3.4). The Schur-complement strategy of eliminating auxiliary degrees of freedom has been employed before, e.g., in Ref. [158], but we generalize it to a larger class of equations, including equations with nonlinear fluxes or sources (Section 2.3.3). We follow Teukolsky [152] whenever possible and refer to Hesthaven and Warburton [126] for details that have become standard in the DG literature.[3]

(2.1): $-\partial_i \mathcal{F}_\alpha{}^i[u_A, v_A; \boldsymbol{x}] + \mathcal{S}_\alpha[u_A, v_A; \boldsymbol{x}] = f_\alpha(\boldsymbol{x})$

[158]: Fortunato, Rycroft, and Saye (2019), *Efficient Operator-Coarsening Multigrid Schemes for Local Discontinuous Galerkin Methods*

[152]: Teukolsky (2016), *Formulation of discontinuous Galerkin methods for relativistic astrophysics*

[126]: Hesthaven and Warburton (2008), *Nodal Discontinuous Galerkin Methods*

3: Reference [152] underpins the *hyperbolic* DG formulations in the `SpECTRE` code. Formulating elliptic and hyperbolic DG schemes in a similar way allows us to share some of the DG implementation details.

[8]: Vincent, Pfeiffer, and **Fischer** (2019), *hp-adaptive discontinuous Galerkin solver for elliptic equations in numerical relativity*

[142]: Kidder et al. (2017), *SpECTRE: A Task-based Discontinuous Galerkin Code for Relativistic Astrophysics*

[152]: Teukolsky (2016), *Formulation of discontinuous Galerkin methods for relativistic astrophysics*

2.3.1 Domain decomposition

We adopt the same domain decomposition based on deformed cubes detailed in Refs. [8, 142, 152] and summarize it here.

A d-dimensional computational domain $\Omega \subset \mathbb{R}^d$ is composed of *elements* $\Omega_k \subset \Omega$ such that $\Omega = \bigcup_k \Omega_k$. Elements do not overlap, but they share boundaries, as illustrated in Fig. 2.1a. Each element carries an invertible map $\boldsymbol{\xi}(\boldsymbol{x})$ from the coordinates $\boldsymbol{x} \in \Omega_k$, in which the elliptic equations (2.1) are formulated, to *logical* coordinates $\boldsymbol{\xi} \in [-1,1]^d$ representing a d-dimensional reference cube. Inversely, $\boldsymbol{x}(\boldsymbol{\xi})$ maps the reference cube to the element Ω_k. We define its Jacobian as

$$\mathrm{J}^i_j \coloneqq \frac{\partial x^i}{\partial \xi^j} \tag{2.8}$$

with determinant J and inverse $(\mathrm{J}^{-1})^j_i = \partial \xi^j / \partial x^i$.

Within each element Ω_k we choose a set of $N_{k,i}$ grid points in every dimension i. We place them at logical coordinates ξ_{p_i}, where the index $p_i \in \{1, \dots, N_{k,i}\}$ identifies the grid point along dimension i. The points are laid out in a regular grid along the logical coordinate axes, so an element has a total of $N_k = \prod_{i=1}^{d} N_{k,i}$ d-dimensional grid points $\boldsymbol{\xi}_p = (\xi_{p_1}, \dots, \xi_{p_d})$. The index $p \in \{1, \dots, N_k\}$ identifies the grid point regardless of dimension. The full domain has $N_{\text{points}} = \sum_k N_k$ grid points. The grid points within each element are not uniformly spaced in logical coordinates. Instead, we choose Legendre-Gauss-Lobatto (LGL) collocation points, i.e., the points ξ_{p_i} fall at the roots of the $(N_{k,i}-1)$th Legendre polynomial plus a point on each side of the element, at -1 and 1. It is equally possible to choose Legendre-Gauss (LG) collocation points, i.e., the roots of the $N_{k,i}$th Legendre polynomial.[4] Figure 2.1b illustrates the geometry of an element.

4: See, e.g., Algorithm 25 in Kopriva [159] for LGL collocation points and Algorithm 23 for LG collocation points.

Fields are represented numerically by their values at the grid points. To facilitate this we construct the one-dimensional Lagrange polynomials

$$\ell_{p_i}(\xi) \coloneqq \prod_{\substack{q_i=1 \\ q_i \neq p_i}}^{N_{k,i}} \frac{\xi - \xi_{q_i}}{\xi_{p_i} - \xi_{q_i}} \quad \text{with} \quad \xi \in [-1, 1] \tag{2.9}$$

and employ their product to define the d-dimensional basis functions

$$\psi_p(\boldsymbol{\xi}) \coloneqq \prod_{i=1}^{d} \ell_{p_i}(\xi^i) \quad \text{with} \quad \boldsymbol{\xi} \in [-1, 1]^d. \tag{2.10}$$

The choice of Lagrange polynomials makes Eq. (2.10) a nodal basis with the useful property $\psi_p(\boldsymbol{\xi}_q) = \delta_{pq}$.[5] We use the nodal basis (2.10) to approximate any field $u(\boldsymbol{x})$ within an element Ω_k by its discretization

5: Lagrange polynomials are illustrated in Fig. 1.18.

$$u^{(k)}(\boldsymbol{x}) \coloneqq \sum_{p=1}^{N_k} u_p \psi_p(\boldsymbol{\xi}(\boldsymbol{x})) \quad \text{with} \quad \boldsymbol{x} \in \Omega_k, \tag{2.11}$$

where the coefficients $u_p = u(\boldsymbol{x}(\boldsymbol{\xi}_p))$ are the field values at the grid points. We denote the set of discrete field values within an element Ω_k as

$$\underline{u}^{(k)} = (u_1, \dots, u_{N_k}), \tag{2.12}$$

and the collection of discrete field values over *all* elements as $\underline{u}$. The discretization (2.11) approximates fields with polynomials of degree $(N_{k,i}-1)$ in dimension i. Although rarely needed, field values at other points within an element can be obtained by Lagrange interpolation (2.11). The field values at element boundaries are double valued because the Lagrange interpolation from neighboring elements to their shared boundary is double valued. Therefore, field approximations will in general be discontinuous at element boundaries.

The test problems in Section 2.4 illustrate a few examples of domain decompositions. We refer the reader to, e.g., Hesthaven and Warburton [126] for further details on the choice of collocation points, basis functions and their relation to spectral properties of DG schemes.

[126]: Hesthaven and Warburton (2008), *Nodal Discontinuous Galerkin Methods*

2.3.2 DG residuals

The DG residuals represent the set of equations to be solved for the discrete primal field values $\underline{u}_A$. The derivation in this section follows the standard procedure, e.g., laid out in Hesthaven and Warburton [126], applied to the generic elliptic flux formulation (2.1), and taking details such as a curved manifold into account.

(2.1): $-\partial_i \mathcal{F}_\alpha{}^i[u_A, v_A; \boldsymbol{x}] + \mathcal{S}_\alpha[u_A, v_A; \boldsymbol{x}] = f_\alpha(\boldsymbol{x})$

In the spirit of a Galerkin scheme we project our target PDEs (2.1) onto the same set of basis functions $\psi_p(\boldsymbol{\xi})$ that is used to approximate fields within an element Ω_k,

$$-\big(\psi_p, \partial_i \mathcal{F}^i\big)_{\Omega_k} + \big(\psi_p, \mathcal{S}\big)_{\Omega_k} = \big(\psi_p, f\big)_{\Omega_k}. \tag{2.13}$$

Here we dropped the index α that enumerates the equations, and we define the inner product on Ω_k,

$$\big(\phi, \pi\big)_{\Omega_k} \coloneqq \int_{\Omega_k} \phi(\boldsymbol{x})\pi(\boldsymbol{x})\sqrt{g}\,\mathrm{d}^d x \tag{2.14a}$$

$$= \int_{[-1,1]^d} \phi(\boldsymbol{x}(\boldsymbol{\xi}))\pi(\boldsymbol{x}(\boldsymbol{\xi}))\sqrt{g}\,\mathrm{J}\,\mathrm{d}^d \xi\,. \tag{2.14b}$$

These integrals are defined with respect to proper volume $\mathrm{d}V = \sqrt{g}\,\mathrm{d}^d x = \sqrt{g}\,\mathrm{J}\,\mathrm{d}^d \xi$, where g denotes the metric determinant in the coordinates $\boldsymbol{x}$ in which Eq. (2.1) is formulated. It refers to the metric that covariant derivatives in the equations are compatible with. Since the basis polynomials, Eq. (2.10), are functions of logical coordinates, we abbreviate $\psi_p(\boldsymbol{\xi}(\boldsymbol{x}))$ with $\psi_p(\boldsymbol{x})$ here.

The terms without derivatives in Eq. (2.13) are straightforward to discretize. We approximate the field f, or similarly $\mathcal{S}$, using the expansion in basis functions (2.11) to find

$$\big(\psi_p, f\big)_{\Omega_k} \approx \big(\psi_p, \psi_q\big)_{\Omega_k} f_q = \mathrm{M}_{pq} f_q, \tag{2.15}$$

using the symmetric *mass matrix* on the element Ω_k,

$$\mathrm{M}_{pq} \coloneqq \big(\psi_p, \psi_q\big)_{\Omega_k} \tag{2.16a}$$

$$= \int_{[-1,1]^d} \psi_p(\boldsymbol{\xi})\psi_q(\boldsymbol{\xi})\sqrt{g}\,\mathrm{J}\,\mathrm{d}^d \xi\,. \tag{2.16b}$$

We will discuss strategies to evaluate the mass matrix on the elements of the computational domain in Section 2.3.6.

The divergence term in Eq. (2.13) encodes the principal part of the elliptic PDEs and requires more care in its discretization. The derivatives in this term will help us couple grid points across element boundaries. To this end we integrate by parts to obtain a boundary term,

$$\big(\psi_p, \partial_i \mathcal{F}^i\big)_{\Omega_k} = -\big(\partial_i \psi_p, \mathcal{F}^i\big)_{\Omega_k} + \big(\psi_p, n_i \mathcal{F}^i\big)_{\partial\Omega_k}, \tag{2.17}$$

where n_i is the outward-pointing unit normal one form on the element boundary $\partial\Omega_k$. The *unnormalized* face normal is computed from the Jacobian as $\tilde{n}_i = \mathrm{sgn}(\xi^j)(\mathrm{J}^{-1})^j_i$, where ξ^j is the logical coordinate that is constant on the particular face and no sum over j is implied. The face

normal is normalized as $n_i = \tilde{n}_i / \sqrt{\tilde{n}_k \tilde{n}_l g^{kl}}$ using the inverse metric $g^{ij}(\boldsymbol{x})$. The surface integral in Eq. (2.17) is defined just like Eq. (2.14),

$$\begin{aligned} \left(\phi, \pi\right)_{\partial\Omega_k} &\coloneqq \int_{\partial\Omega_k} \phi(\boldsymbol{x}) \pi(\boldsymbol{x}) \sqrt{g^\Sigma}\, \mathrm{d}^{d-1}x && \text{(2.18a)} \\ &= \int_{[-1,1]^{d-1}} \phi(\boldsymbol{x}(\boldsymbol{\xi})) \pi(\boldsymbol{x}(\boldsymbol{\xi})) \sqrt{g^\Sigma}\, \mathrm{J}^\Sigma\, \mathrm{d}^{d-1}\xi\,, && \text{(2.18b)} \end{aligned}$$

using the element boundary's $(d-1)$-dimensional proper volume $\mathrm{d}\Sigma = \sqrt{g^\Sigma}\,\mathrm{d}^{d-1}x = \sqrt{g^\Sigma}\,\mathrm{J}^\Sigma\,\mathrm{d}^{d-1}\xi$, where g^Σ is the surface metric determinant induced by the metric g_{ij} and J^Σ is the surface Jacobian.

The crucial step that couples grid points across element boundaries follows from the field $n_i \mathcal{F}^i$ being double valued on any section of the boundary that an element shares with a neighbor, with one value arising from either side. We must make a choice how to combine the two values from either side of a shared element boundary. This choice is often referred to as a *numerical flux*. For now we will denote the function that combines values from both sides of a boundary as $(n_i \mathcal{F}^i)^*$ and refer to Section 2.3.4 for details on our particular choice of numerical flux. Substituting the numerical flux in Eq. (2.17) yields the *weak* form of the equations,

$$\left(\psi_p, \partial_i \mathcal{F}^i\right)_{\Omega_k} = -\left(\partial_i \psi_p, \mathcal{F}^i\right)_{\Omega_k} + \left(\psi_p, (n_i \mathcal{F}^i)^*\right)_{\partial\Omega_k}. \tag{2.19}$$

The numerical flux in Eq. (2.19) introduces a coupling between neighboring elements that allows us to obtain numerical solutions spanning the full computational domain. Another integration by parts of Eq. (2.19) yields the *strong* form of the equations,

$$\left(\psi_p, \partial_i \mathcal{F}^i\right)_{\Omega_k} = \left(\psi_p, \partial_i \mathcal{F}^i\right)_{\Omega_k} + \left(\psi_p, (n_i \mathcal{F}^i)^* - n_i \mathcal{F}^i\right)_{\partial\Omega_k}. \tag{2.20}$$

We will make use of both the strong and the weak form to obtain symmetric DG operators (see Section 2.3.9). Approximating $\mathcal{F}^i$ using its expansion in basis functions (2.11) we find

$$\left(\psi_p, \partial_i \mathcal{F}^i\right)_{\Omega_k} \approx \left(\psi_p, \partial_i \psi_q\right)_{\Omega_k} \mathcal{F}^i_q = \mathrm{MD}_{i,pq} \mathcal{F}^i_q, \tag{2.21}$$

where the *stiffness matrix* on the element Ω_k is

$$\begin{aligned} \mathrm{MD}_{i,pq} &\coloneqq \left(\psi_p, \partial_i \psi_q\right)_{\Omega_k} && \text{(2.22a)} \\ &= \int_{[-1,1]^d} \psi_p(\boldsymbol{\xi})\, \frac{\partial \psi_q}{\partial \xi^j}(\boldsymbol{\xi})\, (\mathrm{J}^{-1})^j_i \sqrt{g}\, \mathrm{J}\, \mathrm{d}^d\xi\,. && \text{(2.22b)} \end{aligned}$$

The divergence term in its weak form can be expressed in terms of the stiffness-matrix transpose $\mathrm{MD}^T_{i,pq} = \mathrm{MD}_{i,qp}$,

$$-\left(\partial_i \psi_p, \mathcal{F}^i\right)_{\Omega_k} \approx -\left(\partial_i \psi_p, \psi_q\right)_{\Omega_k} \mathcal{F}^i_q = -\mathrm{MD}^T_{i,pq} \mathcal{F}^i_q. \tag{2.23}$$

Evaluation of the stiffness matrix and its transpose is discussed in Section 2.3.6.

We now turn towards discretizing the last remaining piece of the DG residuals, the boundary integrals in Eqs. (2.19) and (2.20). It involves a "lifting" operation: the integral only depends on field values on the

element boundary but it may contribute to every component p of the DG residual, hence it is "lifted" to the volume. However, on an LGL grid all components p that correspond to grid points away from the boundary evaluate to zero because they contain at least one Lagrange polynomial that vanishes at the boundary collocation point. This is not the case on an LG grid, where evaluating the Lagrange polynomials on the boundary produces an interpolation into the volume. Expanding the boundary fluxes in basis functions (2.11) we find

$$\left(\psi_p, n_i \mathcal{F}^i\right)_{\partial\Omega_k} \approx \left(\psi_p, \psi_q\right)_{\partial\Omega_k} (n_i \mathcal{F}^i)_q = \mathbb{ML}_{pq} \, (n_i \mathcal{F}^i)_q, \tag{2.24}$$

where we have defined the *lifting operator* on the element Ω_k,

$$\mathbb{ML}_{pq} \coloneqq (\psi_p, \psi_q)_{\partial\Omega_k} \tag{2.25a}$$

$$= \int_{[-1,1]^{d-1}} \psi_p(\xi)\psi_q(\xi)\sqrt{g^\Sigma} \, \mathrm{J}^\Sigma \, \mathrm{d}^{d-1}\xi \, . \tag{2.25b}$$

Section 2.3.6 provides details on evaluating the lifting operator.

Assembling the pieces of the discretization and restoring the index α that enumerates the equations, the DG residuals on the element Ω_k in strong form are

$$-\mathbb{MD}_i \cdot \mathcal{F}_\alpha^{\ i} - \mathbb{ML} \cdot \left((n_i \mathcal{F}_\alpha^{\ i})^* - n_i \mathcal{F}_\alpha^{\ i}\right) + \mathbb{M} \cdot \mathcal{S}_\alpha = \mathbb{M} \cdot f_\alpha, \tag{2.26a}$$

where $\cdot$ denotes a matrix multiplication with the field values over the computational grid of an element. The DG residuals in weak form are

$$\mathbb{MD}_i^T \cdot \mathcal{F}_\alpha^{\ i} - \mathbb{ML} \cdot (n_i \mathcal{F}_\alpha^{\ i})^* + \mathbb{M} \cdot \mathcal{S}_\alpha = \mathbb{M} \cdot f_\alpha. \tag{2.26b}$$

We can choose either the strong or the weak form for each variable α.

Since the fluxes and sources are computed from the primal and auxiliary variables, the DG residuals (2.26) are algebraic equations for the discrete values $\underline{u}_A$ and $\underline{v}_A$ on all elements and grid points in the computational domain. The left-hand side of Eq. (2.26) is an operator $\mathcal{A}(\underline{u}_A, \underline{v}_A)$ and the right-hand side of Eq. (2.26) is a fixed value at every grid point, so Eq. (2.26) has the structure

$$\mathcal{A}(\underline{u}_A, \underline{v}_A) = \underline{b}. \tag{2.27}$$

If the fluxes and sources are linear, the DG operator $\mathcal{A}(\underline{u}_A, \underline{v}_A)$ can be represented as a square matrix, and Eq. (2.27) is a matrix equation. The size of the DG operator $\mathcal{A}(\underline{u}_A, \underline{v}_A)$ is the product of N_{points} with the number of both primal and auxiliary variables.

Figure 2.2 presents a visualization of the DG operator $\mathcal{A}(\underline{u}_A, \underline{v}_A)$ for a Poisson equation on a regular grid. The axes annotate entries of the operator that correspond to the "input" variables v_i and u, and to the corresponding "output" DG residuals. The mass matrix applied to v_i appears as a diagonal line (see Section 2.3.6) and the stiffness matrices applied to both v_i and u appear as block-diagonal and shaded regions for derivatives in x and y, respectively. The remaining entries represent the coupling between neighboring elements through the numerical flux (see Section 2.3.4). Note that the elements Ω_1 and Ω_4 as well as Ω_2 and Ω_3 decouple, because they share no boundaries as they are placed diagonally

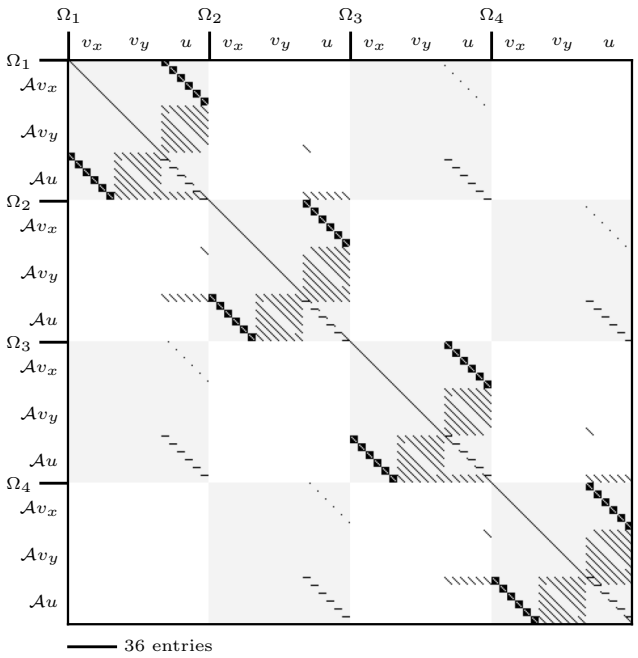


Figure 2.2: Matrix representation of the noncompact DG operator in strong form (2.26a) for a two-dimensional Poisson equation (2.3). The computational domain is partitioned into 2×2 elements with 6×6 LGL grid points each. The image shows the nonzero entries of the operator matrix, i.e., its sparsity pattern, in the order laid out in Fig. 2.1b.

across the 2×2 grid of elements. Solving the Poisson equation amounts to inverting the matrix pictured in Fig. 2.2. However, it is significantly cheaper to invert the equivalent compact operator pictured in Fig. 2.3, which we derive in the following section.

2.3.3 Eliminating auxiliary degrees of freedom

So far we have treated the primal and the auxiliary equations of the first-order formulation on the same footing, which means the discretized DG operator applies to the primal variables as well as to the auxiliary variables. However, the auxiliary equations inflate the size of the operator significantly, increasing both its memory usage and the computational cost for solving it. In this section we eliminate the auxiliary degrees of freedom from the DG operator, demoting them to quantities that are only computed temporarily.

Many publications on DG formulations adopt a "primal formulation" to eliminate auxiliary degrees of freedom from the DG operator.[6] However, in practice we have found a simpler approach taking a Schur complement of the discretized equations in flux form, e.g., applied in Ref. [158], more suited to the generic implementation of DG schemes. The resulting DG operator remains equivalent to the original operator; i.e., it has the same solutions up to numerical precision. This strategy is facilitated by the auxiliary equations defining the auxiliary variables v_A, such as Eqs. (2.3a) and (2.5a). We assume here that the auxiliary fluxes depend only on the primal variables, $\mathcal{F}_{v_A}{}^i = \mathcal{F}_{v_A}{}^i[u_A; \boldsymbol{x}]$, and the auxiliary sources have the form

$$\mathcal{S}_{v_A} = v_A + \tilde{\mathcal{S}}_{v_A}[u_A; \boldsymbol{x}], \tag{2.28}$$

where $\tilde{\mathcal{S}}_{v_A}$ depends only on the primal variables. We further assume $f_{v_A} = 0$ for convenience. All elliptic systems that we consider in this

6: See, e.g., Section 7.2.2 in Hesthaven and Warburton [126] or Section 3 in Arnold et al. [147] for derivations of primal formulations for the Poisson equation.

[126]: Hesthaven and Warburton (2008), *Nodal Discontinuous Galerkin Methods*

[147]: Arnold et al. (2002), *Unified Analysis of Discontinuous Galerkin Methods for Elliptic Problems*

[158]: Fortunato, Rycroft, and Saye (2019), *Efficient Operator-Coarsening Multigrid Schemes for Local Discontinuous Galerkin Methods*

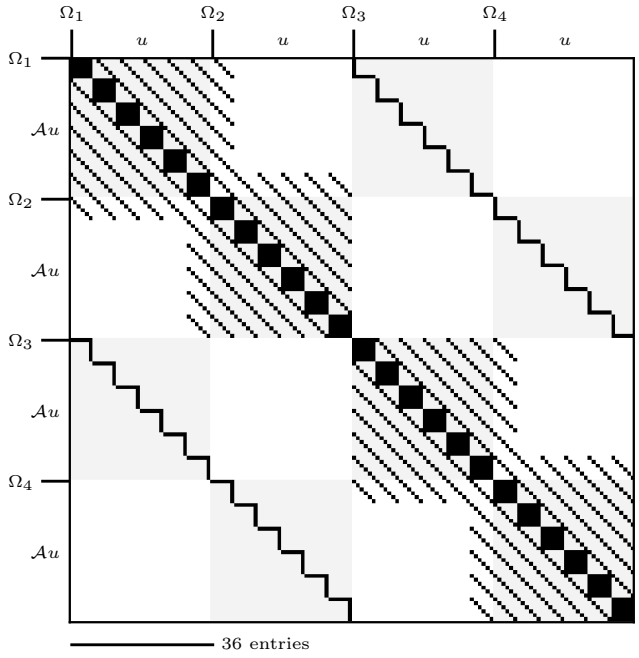


Figure 2.3: Matrix representation of the compact DG operator in strong form (2.30a) for the two-dimensional Poisson problem detailed in Fig. 2.2. No auxiliary degrees of freedom inflate the size of the operator. This matrix is the Schur complement to the matrix pictured in Fig. 2.2.

article fulfill these assumptions. We insert Eq. (2.28) into the strong DG residuals (2.26a) and solve for v_A by inverting the mass matrix to find

$$v_A = \mathrm{D}_i \cdot {\mathcal{F}_{v_A}}^i + \mathrm{L} \cdot \left((n_i {\mathcal{F}_{v_A}}^i)^* - n_i {\mathcal{F}_{v_A}}^i \right) - \tilde{\mathcal{S}}_{v_A}, \tag{2.29}$$

where we define $\mathrm{D}_i := \mathrm{M}^{-1}\mathrm{M}\mathrm{D}_i$ and $\mathrm{L} := \mathrm{M}^{-1}\mathrm{M}\mathrm{L}$. Note that the right-hand side of Eq. (2.29) depends only on the primal variables u_A. Evaluating the DG residuals now amounts to first computing the auxiliary variables v_A by Eq. (2.29), and then using them to evaluate the primal equations. This approach preserves the freedom to use either the strong form (2.26a) for the primal equations,

$$-\mathrm{M}\mathrm{D}_i \cdot {\mathcal{F}_{u_A}}^i - \mathrm{M}\mathrm{L} \cdot \left((n_i {\mathcal{F}_{u_A}}^i)^* - n_i {\mathcal{F}_{u_A}}^i \right) + \mathrm{M} \cdot \mathcal{S}_{u_A} = \mathrm{M} \cdot f_{u_A}, \tag{2.30a}$$

or the weak form (2.26b),

$$\mathrm{M}\mathrm{D}_i^T \cdot {\mathcal{F}_{u_A}}^i - \mathrm{M}\mathrm{L} \cdot (n_i {\mathcal{F}_{u_A}}^i)^* + \mathrm{M} \cdot \mathcal{S}_{u_A} = \mathrm{M} \cdot f_{u_A}. \tag{2.30b}$$

The strong scheme is slightly easier to implement because the primal and auxiliary equations involve the same set of operators. The strong-weak scheme, i.e., selecting the strong form for the auxiliary equations (2.29) and the weak form for the primal equations (2.30b), has the advantage that the DG operator can be symmetric as discussed in Section 2.3.9.

For linear equations the strategy employed in Eq. (2.29) of eliminating the auxiliary variables is equivalent to taking a Schur complement of the DG operator with respect to the (invertible) mass matrix, but the strategy works for nonlinear equations as well. The result is an operator $\mathcal{A}(\underline{u}_A)$ of only the discrete primal variables on all elements and grid points in the computational domain, so the DG residuals (2.30) have the form

$$\mathcal{A}(\underline{u}_A) = \underline{b}. \tag{2.31}$$

The size of the DG operator $\mathcal{A}(\underline{u}_A)$ is the product of N_{points} with the

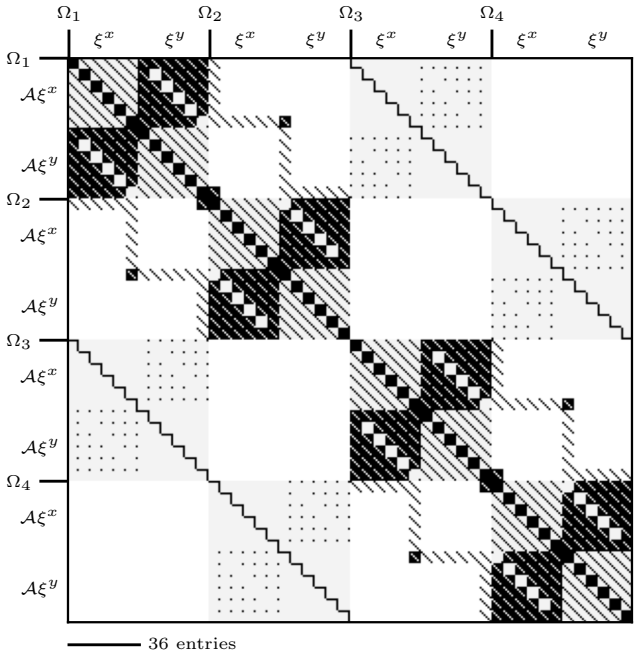


Figure 2.4: Matrix representation of the compact DG operator in strong form (2.30a) for a two-dimensional elasticity problem (2.5) with an isotropic-homogeneous constitutive relation Y^{ijkl}. The computational domain is again partitioned into 2×2 elements with 6×6 LGL grid points each.

number of primal variables. No auxiliary degrees of freedom from the first-order formulation inflate the size of the operator. We refer to such DG operators $\mathcal{A}(\underline{u}_A)$ of only the primal degrees of freedom as *compact* if they also only involve couplings between nearest-neighbor elements [160]. The coupling between elements is related to the choice of numerical flux $(n_i \mathcal{F}_\alpha{}^i)^*$ and the subject of Section 2.3.4. If the fluxes and sources are linear, $\mathcal{A}(\underline{u}_A)$ can be represented as a square matrix.

[160]: Peraire and Persson (2008), *The Compact Discontinuous Galerkin (CDG) Method for Elliptic Problems*

Figures 2.3 and 2.4 present visualizations of $\mathcal{A}$ for a Poisson equation and an elasticity equation, respectively. The block-diagonal structure in Fig. 2.3 represents the DG-discretized two-dimensional Laplacian on the four elements of the computational domain. The entries that break the block-diagonal structure represent the coupling between nearest-neighbor elements through the numerical flux (Section 2.3.4).

2.3.4 A generalized internal-penalty numerical flux

Up to this point we have not made a choice for the numerical flux $(n_i \mathcal{F}^i)^*$ that combines double-valued field values on element boundaries. The numerical flux is a function of the field values on both sides of the boundary. From the perspective of one of the two adjacent elements we refer to the field values on itself as *interior* and to the field values the neighboring element as *exterior*. Contrary to much of the DG literature we formulate the numerical flux entirely in terms of the primal and auxiliary *boundary flux* quantities $n_i \mathcal{F}_{u_A}{}^i$ and $n_i \mathcal{F}_{v_A}{}^i$ on either side of the boundary instead of the primal and auxiliary variables u_A and v_A. This choice keeps our scheme applicable to the wide range of elliptic problems defined in Section 2.2. The numerical flux presented here is a generalization of the symmetric internal penalty (SIP) scheme that is widely used in the literature [8, 126, 147, 161]. Our *generalized internal-penalty numerical flux* is

$$(n_i \mathcal{F}_{v_A}{}^i)^* = \frac{1}{2}\Big[n_i^{\text{int}} \mathcal{F}_{v_A}{}^i(u_A^{\text{int}}) - n_i^{\text{ext}} \mathcal{F}_{v_A}{}^i(u_A^{\text{ext}})\Big], \tag{2.32a}$$

$$\begin{aligned}(n_i \mathcal{F}_{u_A}{}^i)^* = \frac{1}{2}\Big[&n_i^{\text{int}} \mathcal{F}_{u_A}{}^i\big(\partial_j \mathcal{F}_{v_A}{}^j(u_A^{\text{int}}) - \tilde{\mathcal{S}}_{v_A}(u_A^{\text{int}})\big) \\ &- n_i^{\text{ext}} \mathcal{F}_{u_A}{}^i\big(\partial_j \mathcal{F}_{v_A}{}^j(u_A^{\text{ext}}) - \tilde{\mathcal{S}}_{v_A}(u_A^{\text{ext}})\big)\Big] \\ -\sigma\Big[&n_i^{\text{int}} \mathcal{F}_{u_A}{}^i\big(n_j^{\text{int}} \mathcal{F}_{v_A}{}^j(u_A^{\text{int}})\big) \\ &- n_i^{\text{ext}} \mathcal{F}_{u_A}{}^i\big(n_j^{\text{ext}} \mathcal{F}_{v_A}{}^j(u_A^{\text{ext}})\big)\Big].\end{aligned} \tag{2.32b}$$

The numerical flux for the auxiliary equations, Eq. (2.32a), averages the boundary fluxes of the two adjacent elements. The numerical flux for the primal equations, Eq. (2.32b), is an average augmented with a penalty contribution with parameter σ.

Note that the numerical flux (2.32) involves only the primal fields and their derivatives, and thus is independent of the auxiliary fields altogether, as is typical for internal-penalty schemes. This has the practical advantage that the contribution from either side of the boundary to both the primal and the auxiliary numerical flux in Eqs. (2.29) and (2.30) can be computed early in the algorithm and communicated together, coupling only nearest-neighbor elements and thus making the DG operator compact. If the primal numerical flux (2.32b) depended on the auxiliary fields, evaluating the DG operator (2.30) would require a separate communication once the boundary corrections have been added to the auxiliary equation (2.29), effectively coupling nearest-neighbor elements as well as next-to-nearest-neighbor elements.[7]

DG literature usually assumes that the face normals on either side of the boundary are exactly opposite, $n_i^{\text{ext}} = -n_i^{\text{int}}$. This assumption breaks when the background geometry responsible for the normalization of face normals depends on the dynamic variables, since those are discontinuous across the boundary. All of the elliptic problems that we are expecting to solve in the near future are formulated on a fixed background geometry, but it is useful to distinguish between the interior and exterior face normals nonetheless because the quantity $n_i \mathcal{F}^i$ is cheaper to communicate than $\mathcal{F}^i$. Therefore, we always project an element's boundary fluxes onto the face normal before communicating the quantity.

For a simple flat-space Poisson system (2.3) our generalized internal-penalty numerical flux (2.32) reduces to the canonical SIP,[8]

$$(n_i \mathcal{F}_{v_j}{}^i)^* = n_j^{\text{int}} u^* = \frac{1}{2} n_j^{\text{int}} \big(u^{\text{int}} + u^{\text{ext}}\big) \tag{2.33a}$$

$$\begin{aligned}(n_i \mathcal{F}_u{}^i)^* = n_i^{\text{int}} v^{i*} = \frac{1}{2} n_i^{\text{int}} \big(&\partial_i u^{\text{int}} + \partial_i u^{\text{ext}}\big) \\ &- \sigma\big(u^{\text{int}} - u^{\text{ext}}\big).\end{aligned} \tag{2.33b}$$

As is well studied for the canonical SIP numerical flux, the penalty factor σ is responsible for removing zero eigenmodes and impacts the conditioning of the linear operator to be solved [126, 162]. It scales inversely with the element size h and quadratically with the polynomial degree p, both orthogonal to the element face.[9] Both h and p can be

7: Couplings to next-to-nearest-neighbor elements is a well-known disadvantage of LDG-type ("local discontinuous Galerkin") numerical fluxes and has led to the development of compact schemes such as Ref. [160].

[160]: Peraire and Persson (2008), *The Compact Discontinuous Galerkin (CDG) Method for Elliptic Problems*

8: See Eq. (3.21) in Arnold et al. [147] or Section 7.2 in Hesthaven and Warburton [126].

9: See Shahbazi [162] for sharp results for the optimal penalty factor on triangular and tetrahedral meshes, and Table 3.1 in Hillewaert [163] for a generalization to hexahedral meshes.

[162]: Shahbazi (2005), *An explicit expression for the penalty parameter of the interior penalty method*

[163]: Hillewaert (2013), *Development of the Discontinuous Galerkin Method for High-resolution, Large Scale CFD and Acoustics in Industrial Geometries*

different on either side of the element boundary, so we choose

$$\sigma = C \frac{\left(\max(p^{\text{int}}, p^{\text{ext}}) + 1\right)^2}{\min(h^{\text{int}}, h^{\text{ext}})}, \tag{2.34}$$

where we follow Ref. [163] in choosing the scaling with the polynomial degree p on hexahedral meshes, and we follow Ref. [8] in our definition of the element size $h = 2/\sqrt{\tilde{n}_i \tilde{n}_j g^{ij}}$.[10] Note that h generally varies over the element face on curved meshes or on a curved manifold, and that the min operation in Eq. (2.34) is taken pointwise, so σ also varies over the element face. The remaining *penalty parameter* $C \geq 1$ remains freely specifiable. Note also that we do not need to include a problem-specific scale in the penalty factor, as is done in Refs. [155–157], because the generic numerical flux (2.32b) already includes such scales in the fluxes $\mathcal{F}^i$.

[8]: Vincent, Pfeiffer, and **Fischer** (2019), *hp-adaptive discontinuous Galerkin solver for elliptic equations in numerical relativity*

10: Note that Ref. [8] omits the factor of 2 in the definition of h, which we include so the definition reduces to the canonical element size on rectilinear meshes.

[155]: Hartmann and Houston (2008), *An optimal order interior penalty discontinuous Galerkin discretization of the compressible Navier–Stokes equations*
[156]: Epshteyn and Rivière (2007), *Estimation of penalty parameters for symmetric interior penalty Galerkin methods*
[157]: De Basabe, Sen, and Wheeler (2008), *The interior penalty discontinuous Galerkin method for elastic wave propagation: grid dispersion*

2.3.5 Imposing boundary conditions

The flux formulation allows imposing a wide range of boundary conditions relatively easily "through the fluxes" without the need to treat external boundaries any differently than internal boundaries between neighboring elements. Imposing boundary conditions amounts to specifying the exterior quantities in the numerical flux, Eq. (2.32). This strategy is often referred to as imposing boundary conditions through "ghost" elements. As suggested in, e.g., Hesthaven and Warburton [126], we impose boundary conditions on the *average* of the boundary fluxes to obtain faster convergence. Therefore, on external boundaries, we choose for the exterior quantities in the numerical flux (2.32)

$$(n_i \mathcal{F}_\alpha{}^i)^{\text{ext}} = (n_i \mathcal{F}_\alpha{}^i)^{\text{int}} - 2(n_i \mathcal{F}_\alpha{}^i)^{\text{b}}, \tag{2.35}$$

where we set the quantities $(n_i \mathcal{F}_\alpha{}^i)^{\text{b}}$ according to the boundary conditions at hand. Here we define $n_i^{\text{b}} = n_i^{\text{int}}$, i.e., we choose to compute external boundary fluxes with the face normal pointing *out* of the computational domain. The symmetry between the primal and the auxiliary equations in the first-order flux formulation (2.1) that we employ throughout this article makes this approach of imposing boundary conditions particularly straightforward: a choice of *auxiliary* boundary fluxes $(n_i \mathcal{F}_{v_A}{}^i)^{\text{b}}$ imposes Dirichlet-type boundary conditions and a choice of *primal* boundary fluxes $(n_i \mathcal{F}_{u_A}{}^i)^{\text{b}}$ imposes Neumann-type boundary conditions. The choice between Dirichlet-type and Neumann-type boundary conditions can be made separately for every primal variable u_A and every external boundary face, simply by setting either $(n_i \mathcal{F}_{u_A}{}^i)^{\text{b}}$ or $(n_i \mathcal{F}_{v_A}{}^i)^{\text{b}}$ and setting the remaining boundary fluxes to their interior values $(n_i \mathcal{F}^i)^{\text{b}} = (n_i \mathcal{F}^i)^{\text{int}}$. Note that we neither need to distinguish between primal and auxiliary variables in Eq. (2.35), nor take the choice of Dirichlet-type or Neumann-type boundary conditions into account, but we require only that $(n_i \mathcal{F}^i)^{\text{b}}$ be chosen appropriately for every variable. Then, the Neumann-type boundary conditions enter the DG residuals directly through the numerical flux in Eq. (2.30), and the Dirichlet-type boundary conditions enter the DG residuals through the numerical flux in Eq. (2.29), which is substituted in Eq. (2.30).

In practice, this setup means we can initialize $(n_i \mathcal{F}_\alpha{}^i)^{\text{b}} = (n_i \mathcal{F}_\alpha{}^i)^{\text{int}}$ for all

variables on a particular external boundary face when preparing to apply the numerical flux, decide which boundary fluxes to modify based on the boundary conditions we wish to impose on the particular face, and then evaluate Eq. (2.35) to compute the exterior quantities in the numerical flux (2.32). To impose Neumann-type boundary conditions we set the primal boundary fluxes $(n_i \mathcal{F}_{u_A}{}^i)^{\mathrm{b}}$ directly, but to impose Dirichlet-type boundary conditions we typically choose the primal field values u_A^{b} and compute the auxiliary boundary fluxes as $(n_i \mathcal{F}_{v_A}{}^i)^{\mathrm{b}} = n_i^{\mathrm{int}} \mathcal{F}_{v_A}{}^i(u_A^{\mathrm{b}})$.

The auxiliary (Dirichlet-type) external boundary fluxes may depend on the interior primal fields u_A^{int}, and the primal (Neumann-type) external boundary fluxes may depend on both the interior primal fields u_A^{int} as well as the interior auxiliary fields v_A^{int}. This means we can impose a wide range of boundary conditions that may depend linearly or nonlinearly on the dynamic fields. For example, a Robin boundary condition for the Poisson equation (2.2) or (2.6),

$$a\, u + b\, n^i \partial_i u = g(\boldsymbol{x}) \quad \text{on } \partial\Omega, \tag{2.36}$$

where a and b are constants and $g(\boldsymbol{x})$ is a function defined on the boundary, can be implemented as Neumann-type for $b \neq 0$,

$$(n_i \mathcal{F}_u{}^i)^{\mathrm{b}} = \frac{1}{b}\big(g(\boldsymbol{x}) - a\, u^{\mathrm{int}}\big), \tag{2.37}$$

and as Dirichlet-type for $b = 0$,

$$(n_i \mathcal{F}_v{}^i)^{\mathrm{b}} = n_i^{\mathrm{int}} \mathcal{F}_v{}^i(u^{\mathrm{b}}) \quad \text{with} \quad u^{\mathrm{b}} = \frac{1}{a} g(\boldsymbol{x}). \tag{2.38}$$

An important consideration is that boundary conditions are generally nonlinear. Even for linear PDEs, such as the Poisson equation, a simple inhomogeneous Dirichlet boundary condition $u^{\mathrm{b}} \neq 0$ introduces a nonlinearity in the DG operator because $\mathcal{A}(0) \neq 0$. Therefore, we always linearize boundary conditions. For nonlinear equations the boundary conditions linearize along with the DG operator and require no special attention (see Section 2.3.10). However, for linear equations the inhomogeneity in the boundary conditions is the only nonlinearity present in the DG operator, so we skip the full linearization procedure. Instead, we contribute the inhomogeneous boundary conditions $\mathcal{A}(0)$ to the fixed sources, leaving only the linearized boundary conditions in the DG operator,

$$\frac{\delta \mathcal{A}}{\delta u}\underline{u} = \underline{b} - \mathcal{A}(0), \tag{2.39}$$

where $\frac{\delta \mathcal{A}}{\delta u}$ is just $\mathcal{A}$ with linearized boundary conditions. Note that this strategy is equivalent to the full linearization procedure described in Section 2.3.10 at $\underline{u} = 0$. In practice, evaluating $\mathcal{A}(0)$ simplifies significantly for linear equations because only the lifted external boundary corrections contribute to it.

2.3.6 Evaluating the mass, stiffness, and lifting matrices

The mass matrix (2.16), the stiffness matrix (2.22), and the lifting operator (2.25) are integrals that must be evaluated on every element of the

computational domain. We evaluate these integrals on the same grid on which we expand the dynamic fields, which amounts to a Gauss-Lobatto quadrature of an order set by the number of collocation points in the element. This strategy is commonly known as *mass lumping*.[11] Employing mass lumping and our choice of nodal basis (2.10), the mass matrix (2.16) evaluates to

$$\mathrm{M}_{pq} \approx \delta_{pq}\ \sqrt{g}\Big|_p\ \mathrm{J}\big|_p \prod_{i=1}^{d} w_{p_i}. \tag{2.40}$$

Here the coefficients w_{p_i} denote the Legendre-Gauss-Lobatto quadrature weights associated with the collocation points ξ_{p_i}, and the geometric quantities $\sqrt{g}$ and J are evaluated directly on the collocation points.[12] Recall from Section 2.3.1 that the index p enumerates grid points in the regular d-dimensional grid and that p_i denotes the grid point's index along the ith dimension. The diagonal mass-lumping approximation (2.40) has the advantage that it is computationally efficient to apply, invert and store since it amounts to a pointwise multiplication over the computational grid. Note that Eq. (2.40) is exact on a rectilinear LG grid with a flat background geometry, and can be made exact on rectilinear LGL grids by including a correction term without increasing the computational cost for applying or inverting it [164]. The quadrature weights w_{p_i} can be cached and reused by all elements with the same number of collocation points in a dimension.

The strong stiffness matrix (2.22) evaluates to

$$\mathbb{M}\mathrm{D}_{i,pq} \approx \mathrm{M}_{pr}\mathrm{D}_{i,rq}, \tag{2.41a}$$

with

$$\mathrm{D}_{i,rq} = \sum_{j=1}^{d} (\mathrm{J}^{-1})^j_i\Big|_r\ \ell'_{q_j}(\xi_{r_j}) \prod_{\substack{k=1\\k\neq j}}^{d} \delta_{q_k r_k}. \tag{2.41b}$$

Here $\ell'_{q_j}(\xi_{r_j})$ are the one-dimensional "logical" differentiation matrices obtained by differentiating the Lagrange polynomials (2.9) and evaluating them at the collocation points.[13] The stiffness matrix is essentially a "massive" differentiation operator that decomposes into one-dimensional differentiation matrices due to the product structure of the basis functions (2.10) on our hexahedral meshes. The one-dimensional logical differentiation matrices can be cached and reused by all elements with the same number of collocation points in a dimension, keeping the memory cost associated with the stiffness operator to a minimum. The weak stiffness matrix can be computed analogously from the transpose of the logical differentiation matrices.

The lifting operator (2.25) evaluates to

$$\mathbb{M}\mathrm{L}_{pq} \approx \mathrm{M}_{pr}\mathrm{L}_{rq}, \tag{2.42a}$$

with

$$\mathrm{L}_{rq} = \delta_{rq} \sum_{i=1}^{d} (\delta_{q_i 1} + \delta_{q_i N_{k,i}}) \frac{1}{w_{q_i}}\ \sqrt{g^{jk}(\mathrm{J}^{-1})^i_j (\mathrm{J}^{-1})^i_k}\Big|_q. \tag{2.42b}$$

It is diagonal and has a contribution from every face of the element. Note

11: This is the approach taken by Teukolsky [152]. See Eq. (3.7) in Ref. [152] for details on the mass-lumped mass matrix on d-dimensional hexahedral elements. Note that Teukolsky [152] absorbs the metric determinant in the dynamic variables.

[152]: Teukolsky (2016), *Formulation of discontinuous Galerkin methods for relativistic astrophysics*

12: See, e.g., Algorithm 25 in Kopriva [159] for details on computing LGL quadrature weights, and Algorithm 23 for LG quadrature weights.

[164]: Teukolsky (2015), *Short note on the mass matrix for Gauss-Lobatto grid points*

13: See, e.g., Hesthaven and Warburton [126] and Algorithm 37 in Kopriva [159] for details on computing the one-dimensional logical differentiation matrix $\ell'_{q_j}(\xi_{r_j})$ from properties of Legendre polynomials.

that each face only contributes to the LGL grid points on the respective face. On LG grids additional interpolation matrices from the face into the volume appear in this expression. Also note that the root in Eq. (2.42b) is simply the magnitude of the unnormalized face normal $\tilde{n}_j$ [152].

[152]: Teukolsky (2016), *Formulation of discontinuous Galerkin methods for relativistic astrophysics*

Recall that the objective of these matrices is to evaluate the compact DG operator (2.30) along with Eq. (2.29) on every element of the computational domain. In practice, neither matrix must be assembled explicitly and stored on the elements: the mass matrix (2.40) reduces to a multiplication over the computational grid, the stiffness matrix (2.41) involves logical one-dimensional differentiation matrices that are shared between elements, and the lifting operation (2.42) reduces to a multiplication over the grid points on the element face. Since both the stiffness and the lifting operation decompose into a mass matrix and a "massless" operation, the same set of operations can be used to evaluate both Eqs. (2.29) and (2.30), and the mass matrix factors out of the DG operator entirely. Nevertheless, we will see in Section 2.3.9 that it is advantageous to keep the mass matrix in the operator (2.30).

2.3.7 A note on dealiasing

The integral expressions discussed in Section 2.3.6 involve geometric quantities that are typically known analytically, namely the Jacobian and the background metric. Limiting the resolution of the integrals to the quadrature order of the elements can make the scheme susceptible to geometric aliasing because these quantities are resolved with limited precision, potentially reducing the accuracy of the scheme on curved meshes or on a curved manifold. To combat geometric aliasing we can, in principle, precompute the matrices on every element at high accuracy, but at a significant memory cost. Precomputing the matrices is possible in elliptic problems because the geometric quantities remain constant. This is different to time-evolution systems that often involve time-dependent Jacobians ("moving meshes"). Alternatively, a number of dealiasing techniques are available to combat geometric aliasing, and also to combat aliasing arising from evaluating other background quantities on the collocation points, i.e., quantities in the PDEs that are independent of the dynamic variables and known analytically [165]. For example, Ref. [8] interpolates data from the primary LGL grid to an auxiliary LG grid, on which the Jacobian is evaluated, to take advantage of the higher-order quadrature. However, these dealiasing techniques can significantly increase the computational cost for applying the DG operator. We have chosen to employ the simple mass-lumping scheme detailed in Section 2.3.6 to minimize the complexity, computational cost, and memory consumption of the DG operator. Detailed studies of cost-efficient dealiasing techniques are a possible subject of future work.

[165]: Mengaldo et al. (2015), *Dealiasing techniques for high-order spectral element methods on regular and irregular grids*

[8]: Vincent, Pfeiffer, and **Fischer** (2019), *hp-adaptive discontinuous Galerkin solver for elliptic equations in numerical relativity*

2.3.8 Mesh refinement

The domain decomposition into elements, each with their own set of basis functions, allows for two avenues to control the resolution: we can split the domain into more and smaller elements (h refinement) or

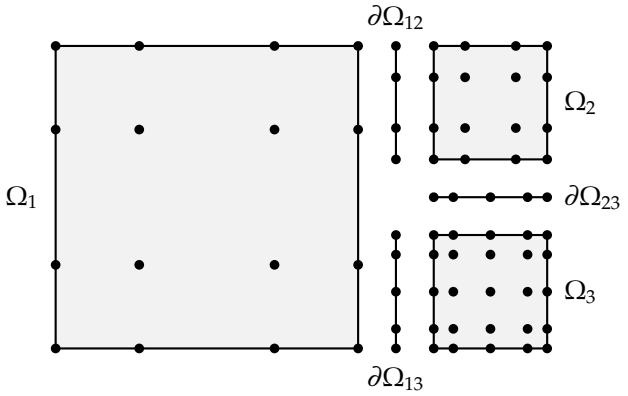


Figure 2.5: A representation of nonconforming element boundaries with mortars in two dimensions. The left element Ω_1 faces two elements Ω_2 and Ω_3 towards the right. Since both Ω_1 and Ω_2 have four vertical grid points their shared mortar $\partial\Omega_{12}$ also has four grid points, but covers only a logical half of Ω_1 (h nonconforming). The element Ω_3 has five vertical grid points, so the mortar $\partial\Omega_{13}$ has $\max(4,5) = 5$ grid points and also covers only a logical half of Ω_1 (hp nonconforming). Elements Ω_2 and Ω_3 are h conforming but differ in their number of horizontal grid points, so their shared mortar $\partial\Omega_{23}$ has $\max(4,5) = 5$ grid points (p nonconforming). Note that the empty space between the elements in this visualization is not part of the computational domain.

increase the number of basis functions within an element (p refinement). We can perform h and p refinement in each dimension independently.

Both h refinement and p refinement can lead to nonconforming boundaries between elements, meaning that grid points on the two sides of the boundary do not coincide. Since we need to work with data from both sides of an element boundary when considering numerical fluxes (see Section 2.3.4) we place *mortars* between elements. A mortar is a $(d-1)$-dimensional mesh that has sufficient resolution to exactly represent discretized fields from both adjacent element faces. Specifically, a mortar $\partial\Omega_{k\bar{k}}$ between the elements Ω_k and $\Omega_{\bar{k}}$ that share a boundary orthogonal to dimension j has $\max(N_{k,i}, N_{\bar{k},i})$ grid points in dimension $i \neq j$. We limit the h refinement of our computational domains such that an element shares its boundary with at most two neighbors per dimension in every direction ("two-to-one balance"). This means a mortar covers either the full element face or a logical half of it in every dimension. Figure 2.5 illustrates an hp-refined scenario with nonconforming element boundaries.

To project field values from an element face to a mortar we employ the $(d-1)$-dimensional *prolongation operator*

$$\mathrm{P}_{\tilde{p}p} = \prod_{i=1}^{d-1} \ell_{p_i}(\tilde{\xi}_{\tilde{p}_i}), \tag{2.43}$$

where p enumerates grid points on the coarser (element face) mesh, $\tilde{p}$ enumerates grid points on the finer (mortar) mesh, and $\tilde{\xi}_{\tilde{p}_i}$ are the coarse-mesh logical coordinates of the fine-mesh collocation points. For mortars that cover the full element face in dimension i the coarse-mesh logical coordinates are just the fine-mesh collocation points, $\tilde{\xi}_{\tilde{p}_i} = \xi_{\tilde{p}_i}$. For mortars that cover the lower or upper logical half of the element face in dimension i they are $\tilde{\xi}_{\tilde{p}_i} = (\xi_{\tilde{p}_i} - 1)/2$ or $\tilde{\xi}_{\tilde{p}_i} = (\xi_{\tilde{p}_i} + 1)/2$, respectively. Note that the prolongation operator (2.43) is just a Lagrange interpolation from the coarser (element face) mesh to the finer (mortar) mesh. The interpolation retains the accuracy of the polynomial approximation because the mortar has sufficient resolution. The prolongation operator is also an L_2 projection (or *Galerkin* projection) because it minimizes the L_2 norm $\int_{\partial\Omega_{k\bar{k}}} (u^{(k)} - u^{(\tilde{k})})^2 \sqrt{g}\, \mathrm{d}^{d-1}x$, where $u^{(\tilde{k})}$ denotes the prolongated

field values on the finer (mortar) mesh.

To project field values from a mortar back to an element face we employ an adjoint R of the prolongation operator such that $\mathrm{RP} = \mathbb{1}$. We also refer to this operation as a *restriction* because it truncates higher modes from the mortar down to the resolution of the element face. Specifically, we employ the mass-conservative adjoint

$$\int_{\partial\Omega_{k\tilde{k}}} \mathrm{R}(u^{(\tilde{k})})\, u^{(k)} \sqrt{g}\, \mathrm{d}^{d-1}x = \int_{\partial\Omega_{k\tilde{k}}} u^{(\tilde{k})} \mathrm{P}(u^{(k)}) \sqrt{g}\, \mathrm{d}^{d-1}x \quad \forall u^{(k)}, u^{(\tilde{k})}. \tag{2.44}$$

In matrix notation the *restriction operator* reduces to

$$\mathrm{R} = \mathrm{M}^{-1}\mathrm{P}^{T}\tilde{\mathrm{M}}, \tag{2.45}$$

where M^{-1} is the inverse mass matrix on the coarser (element face) mesh, $\tilde{\mathrm{M}}$ is the mass matrix on the finer (mortar) mesh, and P^{T} is the transpose of the prolongation operator (2.43).

Note that the d-dimensional restriction and prolongation operators can serve not only to project field values to and from mortars, but also to project field values to and from elements that cover the computational domain at different h- and p-refinement levels. We make no use of projections across refinement levels in this article but will do so in upcoming work for the purpose of adaptive mesh-refinement strategies and for multigrid solvers.[14]

14: See also Sections 3.2 and 3.3 in Ref. [158] for details on the restriction and prolongation operators in the context of multigrid solvers.

[158]: Fortunato, Rycroft, and Saye (2019), *Efficient Operator-Coarsening Multigrid Schemes for Local Discontinuous Galerkin Methods*

2.3.9 A note on symmetry

For practical applications it is often advantageous to work with a symmetric operator. For example, some iterative linear solvers such as conjugate gradients take advantage of the symmetry to invert the operator more efficiently. One can also often show stronger convergence bounds for iterative linear solvers applicable to nonsymmetric matrices, such as GMRES, if the matrix is symmetric [135].

[135]: Saad and Schultz (1986), *GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems*

The compact strong-weak scheme presented in Eq. (2.30b) with the generalized SIP numerical flux (2.32) is symmetric unless the elliptic equations break the symmetry, e.g., with an asymmetric coupling between equations. Note that a curved manifold will typically break the symmetry because it involves first derivatives in Christoffel-symbol contributions to the primal sources [see, e.g., Eq. (2.7)]. It is straightforward to see how the strong-weak scheme can make the DG operator symmetric if the elliptic equations allow it: the strong-weak operator involves a symmetric stiffness term of the schematic form $(\mathrm{MD})^{T}\mathrm{D} = \mathrm{D}^{T}\mathrm{MD}$, whereas the strong scheme has a nonsymmetric expression of the form MDD instead. Note that the "massless" variant of the strong-weak scheme, schematically $\mathrm{M}^{-1}\mathrm{D}^{T}\mathrm{MD}$, is not generally symmetric, and neither is the "massless" strong scheme DD.

2.3.10 Linearizing the operator

To solve nonlinear elliptic equations $\mathcal{A}(\underline{u}) = \underline{b}$ we typically employ a correction scheme, repeatedly solving the linearized equations for a correction quantity $\Delta\underline{u}$. For example, a simple Newton-Raphson correction scheme solves the linearized problem $\frac{\delta\mathcal{A}}{\delta u}(\underline{u})\,\Delta\underline{u} = \underline{b} - \mathcal{A}(\underline{u})$ at fixed $\underline{u}$ and then iteratively corrects $\underline{u} \to \underline{u} + \Delta\underline{u}$. Since the fluxes $\mathcal{F}_\alpha^{\ i}$ are already linear for all elliptic systems we consider, the linearization $\frac{\delta\mathcal{A}}{\delta u}(\underline{u})$ involves only linearizing the sources $\mathcal{S}_\alpha$ and the boundary conditions.

2.3.11 Variations of the scheme

We have made a number of choices to formulate the DG discretization in the preceding sections. This section summarizes some of the choices and presents possible variations to explore in future work.

Massive vs massless scheme We can eliminate the mass matrix in Eq. (2.30) to obtain a "massless" DG operator. However, we have found evidence that iterative linear solvers converge faster when solving the "massive" DG operator. We attribute this behaviour to the symmetry considerations discussed in Section 2.3.9.

Mass lumping We diagonally approximate the mass matrix to reduce the computational cost to apply, invert and store it, and to simplify the scheme (see Section 2.3.6). Dealiasing techniques can potentially increase the accuracy of the scheme on curved meshes as discussed in Section 2.3.7.

LGL vs LG mesh We chose to discretize the DG operator on LGL meshes to take advantage of the collocation points on element boundaries, which simplify computations of boundary corrections. Switching to LG meshes can have the advantage that quadratures are one degree more precise, making the mass-lumping exact on rectilinear grids (see Section 2.3.6).

Numerical flux The generalized internal-penalty numerical flux presented in Section 2.3.4 has proven a viable choice for a wide range of problems so far. However, the ability to switch out the numerical flux is a notable strength of DG methods, and augmenting the numerical flux in the elliptic DG scheme may improve its convergence properties or accuracy. In particular, the choice of penalty, Eq. (2.34), on curved meshes remains a subject of further study.

Strong vs weak formulation We have chosen the strong formulation (2.30a) over the strong-weak formulation (2.30b) because it is slightly simpler and we have, so far, found no evidence that the strong-weak formulation converges faster than the strong formulation, despite the symmetry considerations discussed in Section 2.3.9. However, the strong-weak formulation can be of interest if a symmetric DG operator is necessary, e.g., to take advantage of specialized iterative solvers.

Flux vs primal formulation We have eliminated auxiliary degrees of freedom in the DG operator with a Schur-complement strategy. An alternative strategy is to derive a "primal formulation" of the DG operator (see Section 2.3.3). We have found the flux formulation

easier to implement due to its similarity to hyperbolic DG schemes. Furthermore, Fortunato, Rycroft, and Saye [158] suggest that the flux formulation can be advantageous in conjunction with a multigrid solver.

[158]: Fortunato, Rycroft, and Saye (2019), *Efficient Operator-Coarsening Multigrid Schemes for Local Discontinuous Galerkin Methods*

2.4 Test problems

The following numerical tests confirm the DG scheme presented in this article can solve a variety of elliptic problems. The test problems involve linear and nonlinear systems of PDEs with nonlinear boundary conditions on curved manifolds, discretized on hp-refined domains with curved meshes and nonconforming element boundaries.

For test problems that have an analytic solution we quantify the accuracy of the numerical solutions by computing an L_2 error over all primal variables,

$$\|u - u_{\text{analytic}}\| \coloneqq \left(\frac{\sum_{A,k} \int_{\Omega_k} (u_A - u_{A,\text{analytic}})^2 \, \mathrm{d}V}{\sum_k \int_{\Omega_k} \mathrm{d}V} \right)^{1/2}, \tag{2.46}$$

where the integrals are evaluated with Gauss-Lobatto quadrature on the elements of the computational domain.

To assess the DG operator is functional for our test problems we study the convergence of the discretization error (2.46) under uniform hp refinement of the computational domain (see Section 2.3.8). We compute the h-convergence order under pure uniform h refinement

$$\tau_h \coloneqq \frac{\Delta_h \ln\big(\|u - u_{\text{analytic}}\|\big)}{\Delta_h \ln(h)}, \tag{2.47}$$

where Δ_h denotes the difference between successive h-refinement levels and h is the size of an element. Since we always split elements in half along all logical axes we use $\Delta_h \ln(h) = \ln(2)$. We also compute the exponential convergence scale under pure uniform p refinement

$$\tau_p \coloneqq \Delta_p \log_{10}(\|u - u_{\text{analytic}}\|), \tag{2.48}$$

where Δ_p denotes the difference between successive p-refinement levels.

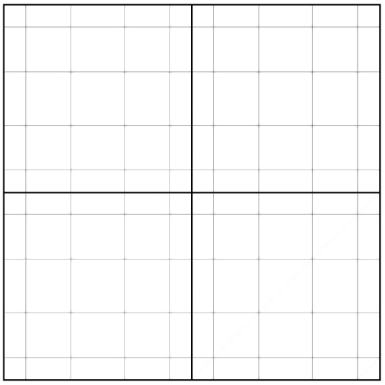


Figure 2.6: The two-dimensional rectilinear domain used in the Poisson problem (Section 2.4.1). Black lines illustrate element boundaries and gray lines represent the LGL grid within each element. This domain is isotropically h refined once, i.e., split once in both dimensions, resulting in four elements. Each element has six grid points per dimension, so fields are represented as polynomials of degree five. This is the domain that Figs. 2.2 and 2.3 are based on, and that is circled in Fig. 2.7.

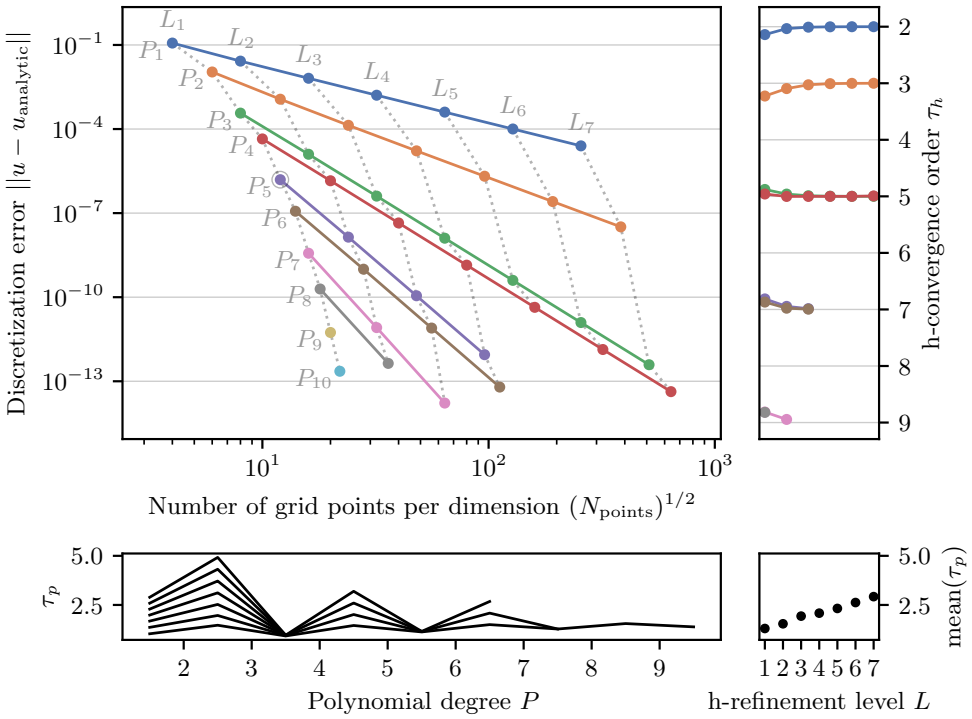


Figure 2.7: Convergence of the two-dimensional Poisson problem detailed in Section 2.4.1 with uniform hp refinement. Solid lines connect numerical solutions where the domain is split into an increasing number of elements (isotropic h refinement), and dotted lines connect numerical solutions with increasing polynomial order (isotropic p refinement). The DG scheme recovers optimal $\mathcal{O}(h^{P+1})$ convergence with odd-order superconvergence under h refinement (right panel) and exponential convergence under p refinement (bottom panels). For reference, the circled configuration is pictured in Fig. 2.6.

2.4.1 A Poisson solution

With this first test problem we establish a simple baseline that the following tests build upon. It is reduced to the absolute essentials to illustrate the basic concepts of the scheme. We solve a flat-space Poisson equation (2.2) in two dimensions for the analytic solution

$$u_{\text{analytic}}(\boldsymbol{x}) = \sin(\pi x)\sin(\pi y) \tag{2.49}$$

on a rectilinear domain $\Omega = [0,1]^2$. The domain is illustrated in Fig. 2.6. To obtain the solution (2.49) numerically we choose the fixed source $f(\boldsymbol{x}) = 2\pi^2 \sin(\pi x)\sin(\pi y)$, select homogeneous Dirichlet boundary conditions $u^{\text{b}} = 0$, and solve the strong compact DG-discretized problem (2.30a) with $C = 1$. This essentially means we invert the matrix depicted in Fig. 2.3 and apply it to the discretization of the fixed source $f(\boldsymbol{x})$. Instead of inverting the matrix directly we employ the iterative elliptic solver of the `SpECTRE` code [10] presented in Ref. [2]. However, note that the technology we use to solve the DG-discretized problem is not relevant for the purpose of this article, since the matrix equation has a unique solution. Assuming the matrix equation is solved to sufficient precision, Eq. (2.46) quantifies the discretization error of the DG scheme.

We solve the problem on a series of uniformly and isotropically refined domains and present the convergence of the discretization error in Fig. 2.7. Under h refinement the scheme recovers optimal $\mathcal{O}(h^{P+1})$ convergence, where P denotes the polynomial degree of the elements. It also recovers

[10]: SpECTRE, spectre-code.org

[2]: **Vu** et al. (2022), *A scalable elliptic solver with task-based parallelism for the* `SpECTRE` *code*. Chapter 3 of this thesis.

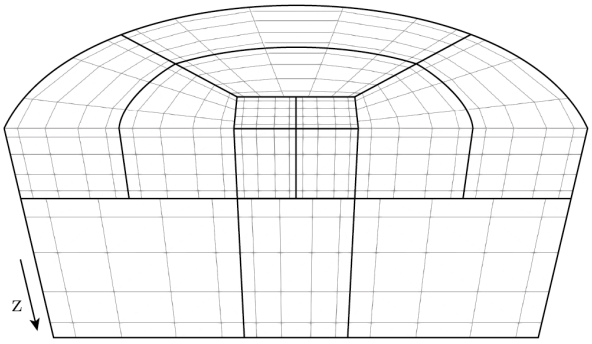


Figure 2.8: A cut through the cylindrical domain used in the elasticity problem (Section 2.4.2). The domain consists of four wedges enveloping a cuboid, and two vertical layers. The layers are partitioned vertically at $z = r_0$ and the cuboid lies radially within $r = r_0$. In the top layer, the wedges are h refined radially once and the cuboid is h refined in the x and y directions once, resulting in 12 elements in the top layer and 5 elements in the bottom layer. Elements in this example have six grid points per dimension, and the wedge-shaped elements have two additional grid points in their angular direction.

the odd-order superconvergence feature expected for the antisymmetric problem (2.49).[15] Under p refinement the scheme recovers exponential convergence. The exponential convergence scale τ_p is modulated by the superconvergence feature and its mean increases linearly with the h-refinement level.

15: See also Fig. 7.9 in Hesthaven and Warburton [126].

2.4.2 Thermal noise in a cylindrical mirror

In this second test problem we solve the equations of linear elasticity (2.4) on a curved mesh with nonconforming element boundaries. The test problem represents a cylindrical mirror that is deformed by pressure from a laser beam incident on one of the sides. This problem arises in studies of Brownian thermal noise in interferometric gravitational-wave detectors [166, 167].[16] Here we consider an analytic solution to this problem that applies in the limit of an isotropic and homogeneous mirror material with constitutive relation

$$Y^{ijkl} = \lambda\, \delta^{ij}\delta^{kl} + \mu \left(\delta^{ik}\delta^{jl} + \delta^{il}\delta^{jk}\right), \tag{2.50}$$

characterized by the Lamé parameter λ and the shear modulus μ, or equivalently by the Poisson ratio $\nu = \lambda/(2(\lambda+\mu))$, Young's modulus $E = \mu(3\lambda + 2\mu)/(\lambda+\mu)$, or the bulk modulus $K = \lambda + 2/3\,\mu$. We assume the material fills the infinite half-space $z \geq 0$, choose a vanishing force density $f^j(x) = 0$, and a Gaussian profile of the laser beam incident at $z = 0$,

$$n_i T^{ij} = n^j \frac{1}{\pi r_0^2} e^{-r^2/r_0^2}. \tag{2.51}$$

Here $T^{ij} = -Y^{ijkl}S_{kl}$ is the *stress*, n_i is the unit normal pointing away from the mirror, i.e., in negative z direction, $r = \sqrt{x^2+y^2}$ is the radial coordinate distance from the axis of symmetry, and r_0 is the beam width. Under these assumptions the displacement field $\xi^i(\boldsymbol{x})$ has the analytic solution [16, 168, 169]

$$u^r = \frac{1}{2\mu}\int_0^\infty \mathrm{d}k\, J_1(kr) e^{-kz}\left(1 - \frac{\lambda+2\mu}{\lambda+\mu} + kz\right)\tilde{p}(k). \tag{2.52a}$$

$$u^z = \frac{1}{2\mu}\int_0^\infty \mathrm{d}k\, J_0(kr) e^{-kz}\left(1 + \frac{\mu}{\lambda+\mu} + kz\right)\tilde{p}(k) \tag{2.52b}$$

[166]: Levin (1998), *Internal thermal noise in the LIGO test masses: A Direct approach*
[167]: Lovelace, Demos, and Khan (2018), *Numerically modeling Brownian thermal noise in amorphous and crystalline thin coatings*

16: See also Section 11.9.2 in Thorne and Blandford [16] for an introduction to the thermal noise problem, and Chapter 5 of this thesis for a detailed study.

[16]: Thorne and Blandford (2017), *Modern Classical Physics*
[168]: Liu and Thorne (2000), *Thermoelastic noise and homogeneous thermal noise in finite sized gravitational wave test masses*
[169]: Lovelace (2007), *The Dependence of test-mass coating and substrate thermal noise on beam shape in the advanced Laser Interferometer Gravitational-Wave Observatory (advanced LIGO)*

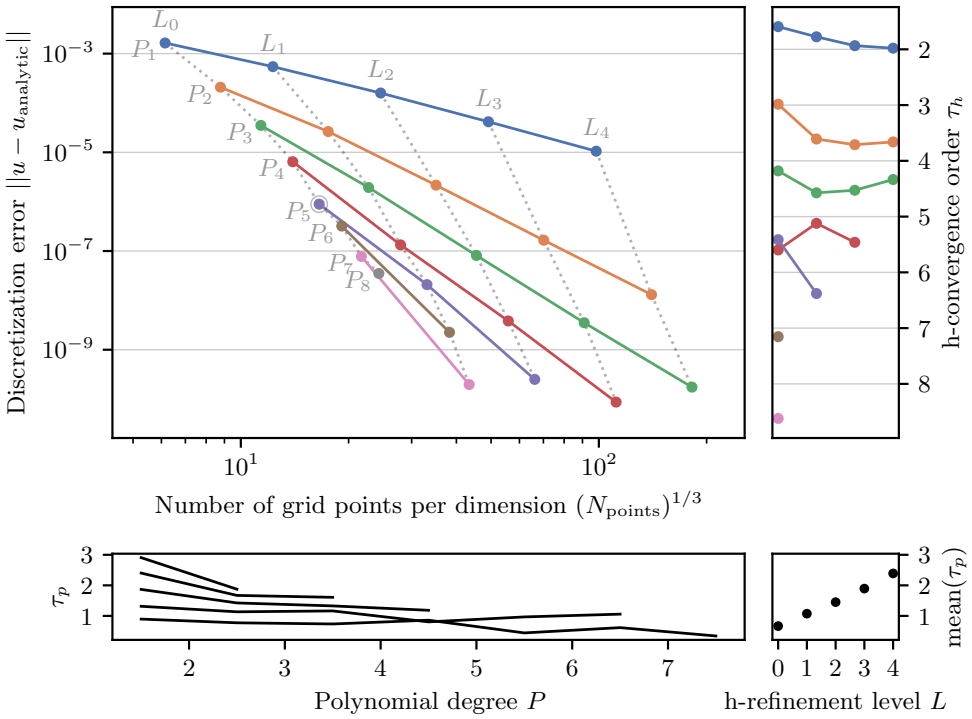


Figure 2.9: Convergence of the three-dimensional elasticity problem detailed in Section 2.4.2 under uniform hp refinement. Plotted is the L_2 error (2.46) over the three components of the displacement field $\xi^i(\boldsymbol{x})$. The refinement is based on the domain pictured in Fig. 2.8 (circled configuration) with curved meshes and nonconforming element boundaries. The DG scheme recovers optimal $\mathcal{O}(h^{P+1})$ convergence under h refinement (right panel) and exponential convergence under p refinement (bottom panels).

and $u^\phi = 0$ in cylindrical coordinates $\{r, \phi, z\}$. Here J_0 and J_1 are Bessel functions of the first kind, and $\tilde{p}(k) = \frac{1}{2\pi} e^{-(kr_0/2)^2}$ is the Hankel transform of the laser-beam profile. We evaluate these integrals numerically at every collocation point in the computational domain to determine the analytic solution.

To obtain numerical solutions to the thermal noise problem we DG discretize the equations of linear elasticity (2.5) on a cylindrical domain with height and radius R, employing the strong compact DG operator (2.30a). Since the stress $T^{ij} = -\mathcal{F}_u{}^{ij}$ is the negative primal flux in the elasticity equations (2.5) we impose Eq. (2.51) as a Neumann-type boundary condition on the base of the cylinder at $z = 0$. We impose the analytic solution (2.52) as Dirichlet-type boundary conditions on the remaining external boundaries of the domain, i.e., on the base at $z = R$ and on the mantle at $r = R$. These boundary conditions mean that we solve for a finite cylindrical section of the infinite half-space analytic solution (2.52). We choose a penalty parameter of $C = 100$ for this problem to eliminate variations in the discretization error arising from curved-mesh contributions to the penalty (2.34) at high resolutions. Table 2.1 summarizes the remaining parameters we use in the numerical solutions.

Table 2.1: Parameters used in the thermal-noise problem (Section 2.4.2). The beam width and the material properties correspond to Table 1 in Lovelace, Demos, and Khan [167]. These material properties characterize a fused-silica mirror, which is a material used in the LIGO gravitational-wave detectors.

Beam width	r_0	177 µm
Outer radius	R	600 µm
Poisson ratio	ν	0.17
Young's modulus	E	72 GPa

Figure 2.8 illustrates the cylindrical domain. It is refined more strongly toward the origin $\boldsymbol{x} = 0$ where the Gaussian laser beam applies pressure. The refinement is both anisotropic and inhomogeneous, leading to nonconforming element boundaries with different polynomial degrees on either side of the boundary, multiple neighbors adjacent to an element face,

or both. Specifically, elements facing the top-layer cuboid or the interface between top and bottom layer are matched two-to-one, and wedge-shaped elements have two additional angular grid points. Therefore, the elements facing the cuboid are both p nonconforming and h nonconforming in the top layer, and p nonconforming in the bottom layer. The elements facing the layer interface are h nonconforming.

Figure 2.9 presents the convergence of the discretization error under uniform hp refinement. Specifically, we split every element in two along all three dimensions to construct additional h-refinement levels, and increment every polynomial degree by one to construct additional p-refinement levels, retaining the nonconforming element boundaries. Note that the wedge-shaped elements retain a higher polynomial degree of $P+2$ along their angular direction throughout the refinement procedure, where P is the polynomial degree of all other elements and dimensions. The DG scheme recovers optimal $\mathcal{O}(h^{P+1})$ convergence under h refinement and exponential convergence under p refinement. Note that the exponential convergence scale τ_p depends on the domain geometry, the structure of the solution, the placement of grid points and the refinement strategy. We have chosen to refine the domain as uniformly as possible here to reliably measure convergence properties of the DG scheme. Optimizing the distribution of elements and grid points with adaptive mesh refinement (AMR) strategies to increase the rate of convergence is the subject of ongoing work.

2.4.3 A black hole in general relativity

Now we apply the DG scheme to solve the Einstein constraint equations of general relativity in the XCTS formulations, which is a set of coupled, nonlinear, elliptic PDEs on a curved manifold (see Section 1.2.3). Solutions to the XCTS equations describe admissible configurations of general-relativistic spacetime and provide initial data for general-relativistic time evolutions.

In this test problem we solve the XCTS equations (1.56) for a Schwarzschild black hole in Kerr-Schild coordinates,

$$\psi = 1, \tag{2.53a}$$

$$\alpha = \left(1 + \frac{2M}{r}\right)^{-1/2}, \tag{2.53b}$$

$$\beta^i = \frac{2M}{r}\alpha^2 l^i, \tag{2.53c}$$

with the background quantities

$$\bar{\gamma}_{ij} = \delta_{ij} + \frac{2M}{r} l_i l_j \tag{2.53d}$$

and

$$K = \frac{2M\alpha^3}{r^2}\left(1 + \frac{3M}{r}\right), \tag{2.53e}$$

where M is the mass parameter, $r = \sqrt{x^2 + y^2 + z^2}$ is the Euclidean coordinate distance, and $l^i = l_i = x^i/r$.[17] The time-derivative quantities

17: See Table 2.1 in Baumgarte and Shapiro [17].

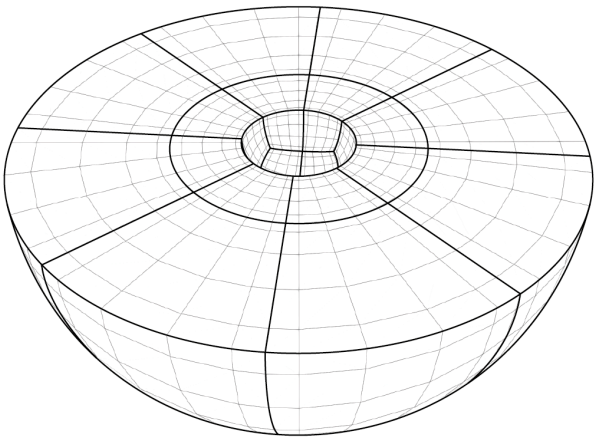


Figure 2.10: A cut through the uniformly refined spherical-shell domain used in the black hole problem (Section 2.4.3). The domain consists of six wedges with a logarithmic radial coordinate map enveloping an excised sphere. In this example each wedge is isotropically h refined once, i.e., split once in all three dimensions, resulting in a total of 48 elements. Note the elements are split in half along their logical axes, so the element size scales logarithmically in radial direction just like the distribution of grid points within the elements. Each element has six grid point per dimension, so fields are represented as polynomials of degree five.

$\bar{u}_{ij}$ and $\partial_t K$ in the XCTS equations (1.56) vanish, as do the matter sources ρ, S, and S^i. Note that we have chosen a conformal decomposition with $\psi = 1$ here, but other choices of ψ and $\bar{\gamma}_{ij}$ that keep the spatial metric $\gamma_{ij} = \psi^4 \bar{\gamma}_{ij}$ invariant are equally admissible.

We solve the XCTS equations numerically for the conformal factor ψ, the product $\alpha\psi$, and the shift β^i. The conformal metric $\bar{\gamma}_{ij}$ and the trace of the extrinsic curvature K are background quantities that are chosen in advance and remain fixed throughout the solve. They are a source of aliasing when evaluated on the computational grid (see Section 2.3.7). Importantly for this test problem the conformal metric $\bar{\gamma}_{ij}$ is not flat, resulting in a problem formulated on a curved manifold. For example, unit normal one forms in the DG scheme are normalized with respect to the conformal metric $\bar{\gamma}_{ij}$ and the metric determinant appears in the mass matrix and in the L_2 error (2.46).

To solve the black hole problem numerically we employ the strong compact DG scheme (2.30a) with $C = 1$ to discretize the XCTS equations (1.56) on a three-dimensional spherical shell, as illustrated in Fig. 2.10. The domain envelops an excised sphere that represents the black hole, so it has an outer and an inner external boundary that require boundary conditions. To obtain the Schwarzschild solution in Kerr-Schild coordinates we impose Eqs. (2.53a) to (2.53c) as Dirichlet-type boundary conditions at the outer boundary of the spherical shell at $r = 10M$. We place the inner radius of the spherical shell at $r = 2M$ and impose nonspinning apparent-horizon boundary conditions at the inner boundary,

$$n^k \partial_k \psi = \frac{\psi^3}{8\alpha} n_i n_j \left((\bar{L}\beta)^{ij} - \bar{u}^{ij} \right) - \frac{\psi}{4} \bar{m}^{ij} \bar{\nabla}_i n_j - \frac{1}{6} K \psi^3, \tag{2.54a}$$

$$\beta^i = -\frac{\alpha}{\psi^2} n^i, \tag{2.54b}$$

where $\bar{m}^{ij} = \bar{\gamma}^{ij} - n^i n^j$. These boundary conditions are not specific to the Schwarzschild solution but ensure the excision surface is an apparent horizon [113].[18] Since the Schwarzschild solution in Kerr-Schild coordinates has an apparent horizon at $r = 2M$ we recover the solution (2.53) when we place the inner radius of the spherical shell at that radius. The apparent-horizon boundary conditions (2.54) do not constrain the lapse α, so we impose Eq. (2.53b) at the inner boundary. The apparent-

[113]: Cook and Pfeiffer (2004), *Excision boundary conditions for black hole initial data*

18: See Section 1.2.3 and note that $n_i = -\bar{s}_i$.

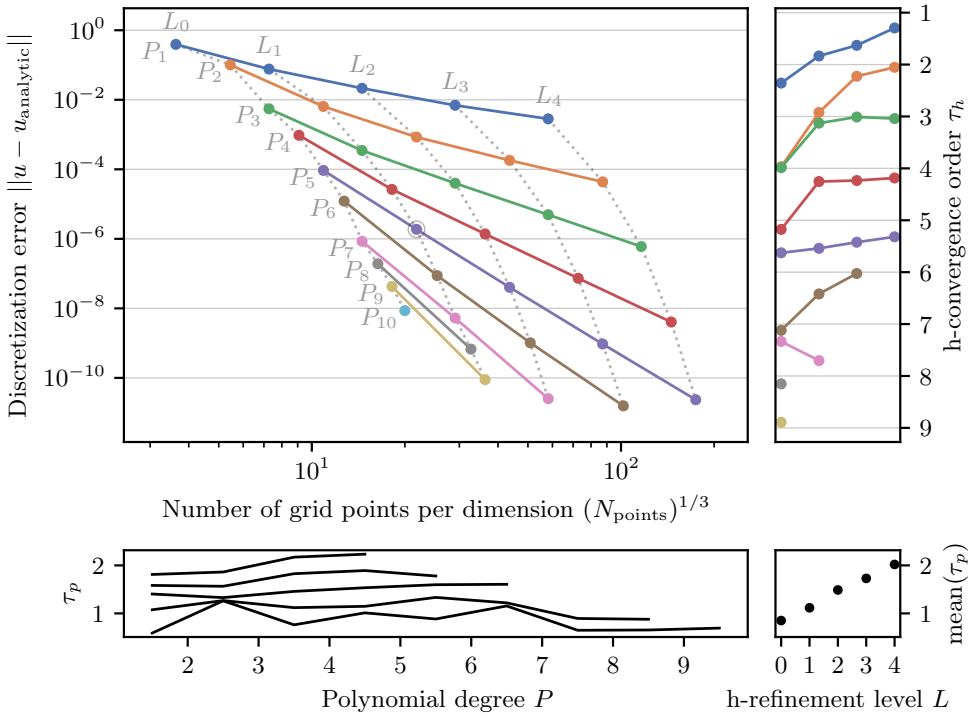


Figure 2.11: Convergence of the three-dimensional black-hole solution with uniform hp refinement. Plotted is the L_2 error (2.46) over all variables of the XCTS equations $\{\psi, \alpha\psi, \beta^i\}$. The circled configuration is pictured in Fig. 2.10. The DG scheme recovers $\mathcal{O}(h^P)$ convergence under h refinement (right panel) and exponential convergence under p refinement (bottom panels).

horizon boundary conditions are of Neumann-type for the variable ψ, of Dirichlet-type for $\alpha\psi$ and β^i, and nonlinear.

Since the XCTS equations (1.56) and the apparent-horizon boundary conditions (2.54) are nonlinear the initial guess for the iterative nonlinear solver becomes relevant. We choose an initial guess close to the analytic solution to ensure fast convergence of the iterative solver to the numerical solution. Note that the initial guess and other details of the iterative solve do not affect the discretization error of the numerical solution once the solve has converged to sufficient precision.

We present the convergence of the discretization error under uniform hp refinement in Fig. 2.11. The DG scheme for the nonlinear black hole problem recovers $\mathcal{O}(h^P)$ convergence under h refinement, which is an order lower than that obtained for the two preceding linear test problems. We find higher-order convergence for pure Dirichlet boundary conditions for this problem, suggesting the apparent-horizon boundary conditions (2.54) are responsible for the reduction of the convergence order. For a Poisson problem with nonlinear boundary conditions, Feistauer, Roskovec, and Sändig [154] also find a loss of convergence under h refinement. Under p refinement the scheme recovers exponential convergence and the mean exponential convergence scale τ_p increases linearly with the h-refinement level.

[154]: Feistauer, Roskovec, and Sändig (2019), *Discontinuous Galerkin method for an elliptic problem with nonlinear Newton boundary conditions in a polygon*

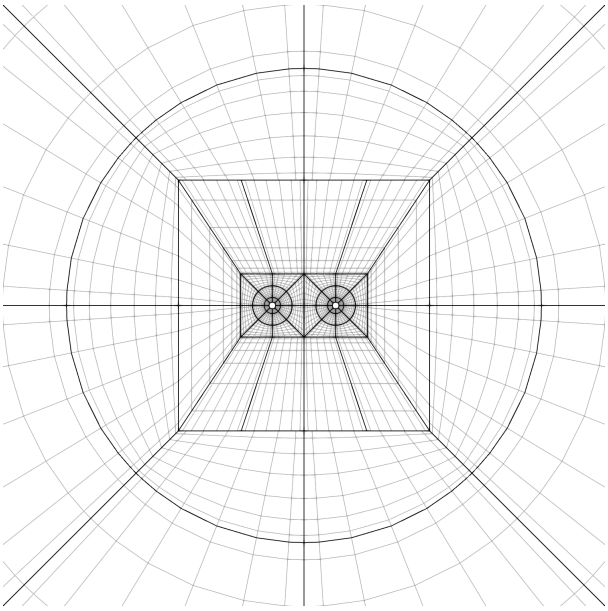


Figure 2.12: A cut through the three-dimensional black-hole binary domain used in Section 2.4.4. It involves two excised spheres centered at $\boldsymbol{C}_n$ along the x axis and extends to a spherical outer surface at radius R. The domain is h refined such that spherical wedges have equal angular size, so the cube-to-sphere boundary is nonconforming. All elements in this picture have eight angular grid points, and $\{7, 8, 8, 9, 11, 11\}$ radial grid points in the layers ordered from outermost to innermost.

2.4.4 A black hole binary

Finally, we solve the Einstein constraint equations in the XCTS formulation as in Section 2.4.3, but now we choose background quantities and boundary conditions that represent two black holes in orbit. This binary black hole problem is of significant relevance in numerical relativity to procure initial data for simulations of merging black holes [17, 74, 75, 78].

[17]: Baumgarte and Shapiro (2010), *Numerical Relativity: Solving Einsteins Equations on the Computer*
[74]: Pfeiffer (2005), *The Initial value problem in numerical relativity*
[75]: Lovelace et al. (2008), *Binary-black-hole initial data with nearly-extremal spins*
[78]: Varma, Scheel, and Pfeiffer (2018), *Comparison of binary black hole initial data sets*

Following the formalism for *superposed Kerr-Schild* initial data, e.g., laid out in Ref. [75, 78], we set the conformal metric and the trace of the extrinsic curvature to the superpositions

$$\bar{\gamma}_{ij} = \delta_{ij} + \sum_{n=1}^{2} e^{-r_n^2/w_n^2} \left(\gamma_{ij}^{(n)} - \delta_{ij}\right) \tag{2.55a}$$

and

$$K = \sum_{n=1}^{2} e^{-r_n^2/w_n^2} K^{(n)}, \tag{2.55b}$$

where $\gamma_{ij}^{(n)}$ and $K^{(n)}$ are the conformal metric and extrinsic-curvature trace of two isolated Schwarzschild black holes in Kerr-Schild coordinates as given in Eqs. (2.53). They have mass parameters M_n and are centered at coordinates $\boldsymbol{C}_n$, with r_n being the Euclidean coordinate distance from either center. The superpositions are modulated by two Gaussians with widths w_n. The time-derivative quantities $\bar{u}_{ij}$ and $\partial_t K$ in the XCTS equations (1.56) vanish, as do the matter sources ρ, S and S^i.

To handle orbital motion we split the shift in a *background* and an *excess* contribution [170],

[170]: Pfeiffer (2003), *Initial Data for Black Hole Evolutions*

$$\beta^i = \beta^i_{\text{background}} + \beta^i_{\text{excess}}, \tag{2.56}$$

and choose the background shift

$$\beta^i_{\text{background}} = (\boldsymbol{\Omega}_0 \times \boldsymbol{x})^i, \tag{2.57}$$

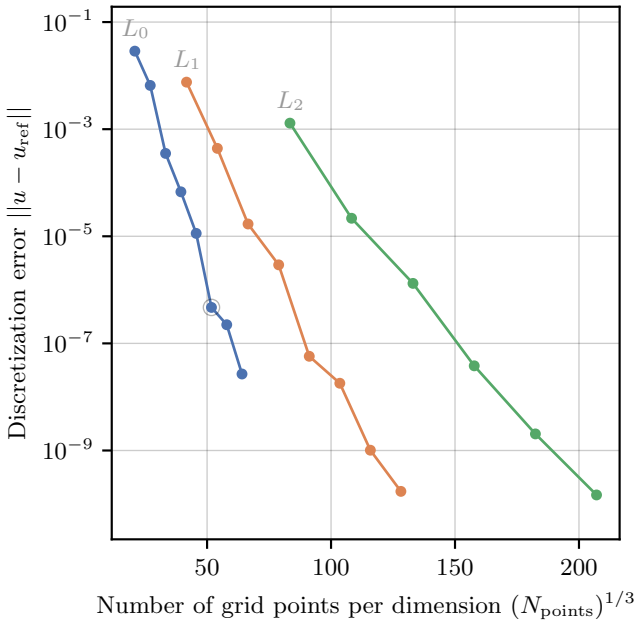


Figure 2.13: Exponential convergence of the three-dimensional black-hole binary problem under uniform p refinement (solid lines) for three uniform h-refinement levels. The circled configuration is pictured in Fig. 2.12. Plotted here is the L_2 error (2.59) over all variables $\{\psi, \alpha\psi, \beta^i_{\mathrm{excess}}\}$ and all three interpolation points $\boldsymbol{x}_m$.

where $\boldsymbol{\Omega}_0$ is the orbital angular velocity. We insert Eq. (2.56) in the XCTS equations (1.56) and henceforth solve them for $\beta^i_{\mathrm{excess}}$, instead of β^i.

We solve the XCTS equations on the domain depicted in Fig. 2.12. It has two excised spheres with radius $2M_n$ that are centered at $\boldsymbol{C}_n$, and correspond to the two black holes, and an outer spherical boundary at finite radius R. We impose boundary conditions on these three boundaries as follows. At the outer spherical boundary we impose asymptotic flatness,

$$\psi = 1, \quad \alpha\psi = 1, \quad \beta^i_{\mathrm{excess}} = 0. \tag{2.58}$$

Since the outer boundary is at a finite radius, the solution will only be approximately asymptotically flat. On the two excision boundaries we impose nonspinning apparent-horizon boundary conditions, Eq. (2.54). For the lapse we choose to impose the isolated solution (2.53b) as Dirichlet conditions at both excision surfaces. Note that this choice differs slightly from Ref. [78], where the *superposed* isolated solutions are imposed on the lapse at both excision surfaces.

[78]: Varma, Scheel, and Pfeiffer (2018), *Comparison of binary black hole initial data sets*

Since the binary black hole problem has no analytic solution we assess the precision of numerical solutions by comparing them to a high-resolution reference configuration. Specifically, we interpolate all five fields $u_A = \{\psi, \alpha\psi, \beta^i_{\mathrm{excess}}\}$ to a set of sample points $\boldsymbol{x}_m$. Then, we compute the discretization error as an L_2 norm of the difference to the high-resolution reference run over all fields and sample points,

$$\|u - u_{\mathrm{ref}}\| := \left(\sum_{A,m} \left(u_A(\boldsymbol{x}_m) - u_{A,\mathrm{ref}}(\boldsymbol{x}_m) \right)^2 \right)^{1/2}. \tag{2.59}$$

Figure 2.13 presents the convergence of the discretization error under uniform hp refinement for our strong compact DG scheme (2.30a) with $C = 1$. Specifically, we obtain h-refinement levels from the domain depicted in Fig. 2.12 by splitting all elements in two along their three

Table 2.2: Sample points $\boldsymbol{x}_m$ used in Eq. (2.59) and the value of the reference solution at the sample points.

	$\boldsymbol{x}_1 = (8.846, 0, 0)$	$\boldsymbol{x}_2 = (0, 0, 0)$	$\boldsymbol{x}_3 = (100, 0, 0)$
ψ	1.0919141	1.0602545	1.0033643
$\alpha\psi$	0.7072066	0.9381658	0.9966282
β^x_{excess}	0.3870172	0	0.0008802
β^y_{excess}	−0.1273493	0	−0.0003467
β^z_{excess}	0	0	0

logical axes. We obtain p-refinement levels by incrementing the number of grid points by one in all elements and dimensions. The DG scheme recovers exponential convergence under p refinement, and suggests the same $\mathcal{O}(h^P)$ convergence under h refinement that we have found for the single black hole problem in Section 2.4.3. We have chosen $M_n = 0.4229$, $\boldsymbol{C}_n = (\pm 8, 0, 0)$, $\Omega_0 = 0.0144$, $w_n = 4.8$, $R = 300$, and sample points along the x axis at $x_1 = 8.846$ (near horizon), $x_2 = 0$ (origin) and $x_3 = 100$ (far field) here. For the high-resolution reference configuration in Eq. (2.59) we use a run that is h refined twice, and has one grid point more per element and dimension than the highest-resolution configuration included in Fig. 2.13. The reference values at the interpolation points are listed in Table 2.2. We have verified that these values are consistent with the same problem solved with the `SpEC` [55, 139] code up to an absolute error of at most 10^{-7}, which is the precision we report in Table 2.2.

[55]: Spectral Einstein Code (SpEC), `black-holes.org/code/SpEC`
[139]: Pfeiffer et al. (2003), *A multidomain spectral method for solving elliptic equations*

In forthcoming work we intend to employ the DG scheme that we have presented here to develop a scalable initial-data solver for binaries involving black holes and neutron stars in the `SpECTRE` numerical relativity code [2].

[2]: **Vu** et al. (2022), *A scalable elliptic solver with task-based parallelism for the `SpECTRE` code*. Chapter 3 of this thesis.

2.5 Conclusion and future work

We have presented a unified discontinuous Galerkin (DG) internal-penalty scheme that is applicable to a wide range of elliptic equations. Our scheme applies to linear and nonlinear second-order elliptic PDEs of one or more variables, where the variables can be scalars, vectors, or tensors of higher rank. It does not require problem-specific modifications of the DG discretization or of the numerical fluxes that couple neighboring elements. The scheme supports a wide range of linear and nonlinear boundary conditions, and applies to equations formulated on curved manifolds. We demonstrate its versatility by solving a simple Poisson problem, a linear elasticity problem on a curved mesh with nonconforming element boundaries, and two nonlinear problems in general relativity involving black holes. The unified DG scheme is capable of solving these problems with no structural changes. It recovers optimal $\mathcal{O}(h^{P+1})$ convergence for the linear test problems and $\mathcal{O}(h^P)$ convergence for the nonlinear test problems, where P is the polynomial degree of the elements. The scheme is implemented in the open-source `SpECTRE` code [10] and the results presented in this article are reproducible with the supplemental input-file configurations [171].

[10]: SpECTRE, spectre-code.org

[171]: Supplemental material: input-file configurations to reproduce the results presented in this article with the SpECTRE code, `arXiv:2108.05826/anc`

The DG scheme developed here can potentially be improved in multiple ways in future work. Dealiasing techniques have the potential to increase the accuracy of the scheme on curved meshes and for equations with background quantities. The choice of penalty on curved meshes remains a subject of ongoing study. Furthermore, detailed studies of the symmetry

of the DG operator and related adjustments to the scheme, such as switching to the strong-weak formulation, can potentially make the DG operator faster to solve.

Since the convergence properties of the DG scheme are sensitive to the specifics of the computational domain, we have chosen to refine the domains as uniformly as possible while retaining some important features, such as curved meshes and nonconforming element boundaries. For practical applications it is typically more important to obtain steep rather than uniform convergence, in order to conserve computational resources and thus achieve faster or more precise solves. Therefore, a focus of future work will be to develop adaptive mesh-refinement strategies for the elliptic DG scheme that place grid points in regions and dimensions of the domain that dominate the discretization error.

Once the DG discretization of the elliptic equations is at hand, numerical techniques for solving the resulting matrix equation become important. Sophisticated linear and nonlinear iterative algorithms are necessary to solve high-resolution elliptic problems in parallel on large computing clusters. Many of the choices we have made in the development of the DG scheme are motivated by such large-scale applications. For this purpose we are developing a scalable multigrid-Schwarz preconditioned Newton-Krylov iterative solver with task-based parallelism that will be presented in [2].

[2]: **Vu** et al. (2022), *A scalable elliptic solver with task-based parallelism for the* `SpECTRE` *code*. Chapter 3 of this thesis.

Acknowledgments

The authors thank Trevor Vincent, Hannes Rüter, Nils Deppe, Will Throwe, Lawrence Kidder, and Saul Teukolsky for helpful discussions. N. F. also thanks the Cornell Center for Astrophysics and Planetary Science and TAPIR at Caltech for the hospitality and financial support during research stays. Computations were performed with the `SpECTRE` code [10] on the Minerva cluster at the Max Planck Institute for Gravitational Physics. The figures in this article were produced with `dgpy` [11], `matplotlib` [172, 173], `TikZ` [174], and `ParaView` [175].

[10]: SpECTRE, spectre-code.org

[11]: dgpy, 10.5281/zenodo.5086181

[172]: Hunter (2007), *Matplotlib: A 2D graphics environment*

[173]: matplotlib, 10.5281/zenodo.4268928

[174]: pgf - A Portable Graphic Format for TeX, `github:pgf-tikz/pgf`

[175]: Ahrens, Geveci, and Law (2005), *`ParaView`: An End-User Tool for Large-Data Visualization*

Task-based parallel elliptic solver in SpECTRE

3

Publication

This chapter is based on the article *A scalable elliptic solver with task-based parallelism for the SpECTRE numerical relativity code* [2], published in *Phys. Rev. D* ***105***, *084027* on Apr 18, 2022 (arXiv:2111.06767). It presents the new elliptic solver that I have developed to solve the discretized elliptic equations developed in Chapter 2, in parallel and on supercomputers. The development of this new, scalable, and highly parallel solver for elliptic problems in numerical relativity, implemented in the next-generation SpECTRE code, represents the majority of my Ph.D. work.

Authors **Nils L. Vu**, Harald P. Pfeiffer, Gabriel S. Bonilla, Nils Deppe, François Hébert, Lawrence E. Kidder, Geoffrey Lovelace, Jordan Moxon, Mark A. Scheel, Saul A. Teukolsky, William Throwe, Nikolas A. Wittek, and Tom Włodarczyk

Abstract Elliptic partial differential equations must be solved numerically for many problems in numerical relativity, such as initial data for every simulation of merging black holes and neutron stars. Existing elliptic solvers can take multiple days to solve these problems at high resolution and when matter is involved, because they are either hard to parallelize or require a large amount of computational resources. Here we present a new solver for linear and nonlinear elliptic problems that is designed to scale with resolution and to parallelize on computing clusters. To achieve this we employ a discontinuous Galerkin discretization, an iterative multigrid-Schwarz preconditioned Newton-Krylov algorithm, and a task-based parallelism paradigm. To accelerate convergence of the elliptic solver we have developed novel subdomain-preconditioning techniques. We find that our multigrid-Schwarz preconditioned elliptic solves achieve iteration counts that are independent of resolution, and our task-based parallel programs scale over 200 million degrees of freedom to at least a few thousand cores. Our new code solves a classic initial data problem for binary black holes faster than the spectral code SpEC when distributed to only eight cores, and in a fraction of the time on more cores. It is publicly accessible in the next-generation SpECTRE numerical relativity code. Our results pave the way for highly parallel elliptic solves in numerical relativity and beyond.

Declaration of authorship I am the lead author who wrote this article, argued the direction it should take, developed the task-based parallel iterative algorithms presented in the article, implemented them in the SpECTRE code, and performed the numerical computations to apply the new elliptic solver to test problems. Harald Pfeiffer acted as advisor in this project and provided editorial feedback on the article. The remaining co-authors contributed to the SpECTRE code in a manner that enabled the present research. For example, Gabriel Bonilla contributed coordinate

maps that are essential to construct the computational domains, Nils Deppe contributed the parallelisation infrastructure that interfaces with `Charm++`, and Jordan Moxon contributed the Morton ("z-order") space-filling curve used to distribute elements among cores. The contributions by all authors were essential to conduct the present research, and highlight the collaborative research effort with the open-source `SpECTRE` code. The development of the new elliptic solver based on the foundations provided by the `SpECTRE` code, and hence the subject of the present article, is entirely my own work.

3.1 Introduction

Solving elliptic partial differential equations (PDEs) numerically is important in many areas of science, including numerical relativity [17]. All numerical time evolutions begin with initial data that capture the physical scenario to be evolved, and the initial data must typically satisfy a set of constraint equations formulated as elliptic PDEs. Specifically, to construct initial data for general-relativistic simulations of black holes and neutron stars we must solve the Einstein constraint equations, which admit formulations as elliptic PDEs [74, 176]. Binary configurations of black holes and neutron stars enjoy particular prominence as primary sources for gravitational-wave detectors, and numerical simulations of these systems play an essential role in their observations [18, 32–34, 60, 177].

To construct initial data for general-relativistic simulations, the numerical relativity (NR) community has put considerable effort towards developing numerical codes that solve elliptic problems. Most of the existing codes employ spectral methods to discretize the elliptic equations, such as `LORENE` [178, 179] and `TwoPunctures` [88, 180], `Spells` [109, 112, 139, 181, 182] that is part of `SpEC` [55], as well as `SGRID` [107, 183], `KADATH` [89, 133] and `Elliptica` [108]. The `COCAL` [110, 184] code employs finite-difference methods, and `NRPyElliptic` [185] a hyperbolic relaxation scheme [186]. All of these codes vary significantly in the numerical methods employed to solve the discretized equations. For example, `SpEC` uses the `PETSc` library to perform an iterative matrix-free solve with a custom preconditioner [139], whereas `KADATH` and `Elliptica` construct explicit matrix representations and invert the matrices directly [108, 133].

While successful in constructing initial data for many general-relativistic scenarios, these codes can still take a significant amount of time or require excessive computational resources to solve the elliptic problems. For example, the `SpEC` and `SGRID` codes typically require a few hours to days to solve for initial data that involves orbiting neutron stars, at a resolution required for state-of-the-art simulations, using $\mathcal{O}(10)$ cores for the computation. `KADATH`, on the other hand, quote a few hours to solve for low-resolution initial data involving orbiting neutron stars on about 128 cores, and "a larger timescale" and more cores for higher resolutions, with high memory demands for the explicit matrix construction [89, 133], and assuming symmetry with respect to the orbital plane [89]. `Elliptica` also quote a few days to solve an initial data problem for a black hole–neutron star binary (BHNS) on 20 cores [108].

[17]: Baumgarte and Shapiro (2010), *Numerical Relativity: Solving Einsteins Equations on the Computer*

[74]: Pfeiffer (2005), *The Initial value problem in numerical relativity*
[176]: Cook (2000), *Initial Data for Numerical Relativity*

[178]: LORENE, `http://www.lorene.obspm.fr`
[179]: Grandclement (2006), *Accurate and realistic initial data for black hole-neutron star binaries*
[88]: Ansorg, Brügmann, and Tichy (2004), *A Single-domain spectral method for black hole puncture data*
[180]: Ansorg (2005), *A Double-domain spectral method for black hole excision data*
[109]: Tacik et al. (2015), *Binary Neutron Stars with Arbitrary Spins in Numerical Relativity*
[112]: Tacik et al. (2016), *Initial data for black hole–neutron star binaries, with rotating stars*
[139]: Pfeiffer et al. (2003), *A multidomain spectral method for solving elliptic equations*
[181]: Ossokine et al. (2015), *Improvements to the construction of binary black hole initial data*
[182]: Foucart et al. (2008), *Initial data for black hole-neutron star binaries: A Flexible, high-accuracy spectral method*
[55]: Spectral Einstein Code (SpEC), `black-holes.org/code/SpEC`

[107]: Dietrich et al. (2015), *Binary Neutron Stars with Generic Spin, Eccentricity, Mass ratio, and Compactness - Quasi-equilibrium Sequences and First Evolutions*
[183]: Tichy et al. (2019), *Constructing binary neutron star initial data with high spins, high compactnesses, and high mass ratios*

[89]: Papenfort et al. (2021), *New public code for initial data of unequal-mass, spinning compact-object binaries*
[133]: Grandclément (2010), *KADATH: A spectral solver for theoretical physics*

[108]: Rashti et al. (2021), *Elliptica: a new pseudo-spectral code for the construction of initial data*

[110]: Tsokaros, Uryū, and Rezzolla (2015), *New code for quasiequilibrium initial data of binary neutron stars: Corotating, irrotational, and slowly spinning systems*
[184]: Uryū and Tsokaros (2012), *New code for equilibriums and quasiequilibrium initial data of compact objects*

[185]: Assumpcao, Werneck, Jacques, et al. (2021), *NRPyElliptic: A Fast Hyperbolic Relaxation Elliptic Solver for Numerical Relativity, I: Conformally Flat, Binary Puncture Initial Data*

[186]: Rüter et al. (2018), *Hyperbolic Relaxation Method for Elliptic Equations*

Despite the time required to solve the elliptic initial data problem, simulations of merging black holes and neutron stars are currently dominated by their time evolution, which can take weeks to months. However, significant efforts are underway to develop faster and more accurate evolution codes for next-generation numerical relativity. The open-source `SpECTRE` [10, 142] code aims to evolve general-relativistic multiphysics scenarios on petascale and future exascale computers, and is the main focus of this article. Other recent developments include the `CarpetX` driver for the Einstein Toolkit [187] that is based on the `AMReX` framework [188], and the `Dendro-GR` [189], `Nmesh` [190], `bamps` [191], `GRAthena++` [192], and `ExaHyPE` [193] codes.

[10]: SpECTRE, spectre-code.org
[142]: Kidder et al. (2017), *SpECTRE: A Task-based Discontinuous Galerkin Code for Relativistic Astrophysics*
[187]: CarpetX, 10.5281/zenodo.6131528
[189]: Fernando et al. (2019), *Massively Parallel Simulations of Binary Black Hole Intermediate-Mass-Ratio Inspirals*
[190]: Tichy, Adhikari, and Ji (2020), *Numerical relativity with the new Nmesh code*
[191]: Bugner et al. (2016), *Solving 3D relativistic hydrodynamical problems with weighted essentially nonoscillatory discontinuous Galerkin methods*
[192]: Daszuta et al. (2021), *GRAthena++: Puncture Evolutions on Vertex-centered Oct-tree Adaptive Mesh Refinement*
[193]: Reinarz et al. (2020), *ExaHyPE: An engine for parallel dynamically adaptive simulations of wave problems*

To seed these next-generation evolutions of general-relativistic scenarios with initial data, we have developed a highly scalable elliptic solver based on discontinuous Galerkin methods, matrix-free iterative algorithms, and task-based parallelism. We focus strongly on parallelization to take advantage of the increasing number of cores in high-performance computing (HPC) systems. These systems have at least $\mathcal{O}(10)$, but often closer to 50–100, physical cores per node, often with many thousand interconnected nodes. Therefore, even routine compute jobs that request only a few nodes on contemporary HPC clusters, and hence spend little to no time waiting in a queue, have tens to hundreds of cores at their disposal. Larger compute jobs with thousands of cores and more are also readily available, and the amount of available computational resources is expected to increase rapidly in the future.

The `SpECTRE` code embraces parallelism as a core design principle [10, 142]. It employs a task-based parallelism paradigm instead of the conventional message passing interface (MPI) protocol. MPI-parallelized programs typically alternate between computation and communication at global synchronization points, meaning that all threads must reach a globally agreed-upon state before the program proceeds. Global synchronization points can limit the effective use of the available cores when some threads reach the synchronization later than others, thus holding up the program. The effect becomes more pronounced with increasing core count, often limiting the number of cores that MPI-parallelized programs can efficiently scale to. Task-based parallel programs, on the other hand, aim to avoid global synchronization points as much as possible. They partition the computational work into interdependent tasks and distribute them among the available cores. Tasks can migrate to undersubscribed cores while the program is running to balance the computational load. `SpECTRE` builds upon the `Charm++` [141] task-based parallelism and CPU-abstraction library. Reference [142] describes `SpECTRE`'s task-based parallism paradigm in more detail.

[141]: The Charm++ Parallel Programming System, `https://charm.cs.illinois.edu`

Our new elliptic solver in the `SpECTRE` code is based on the prototype presented in Ref. [8] and employs the discontinuous Galerkin discretization for generic elliptic equations developed in Ref. [1], which makes it applicable to a wide range of elliptic problems in numerical relativity and beyond. In this article we present the task-based iterative algorithms that we have developed to parallelize the elliptic solver effectively on computing clusters, including novel subdomain-preconditioning techniques. We demonstrate that our new elliptic solver can solve a classic initial data problem for binary black holes faster than `SpEC` when running on as few as eight cores, and in a fraction of the time on a computing cluster. In

[8]: Vincent, Pfeiffer, and **Fischer** (2019), *hp-adaptive discontinuous Galerkin solver for elliptic equations in numerical relativity*
[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

particular, the number of iterations that our new elliptic solver requires to converge remains constant with increasing resolution. The additional computational work needed to solve high-resolution problems manifests in subproblems that become either more numerous or more expensive, but that can be solved in parallel to offset the increase in runtime.

This article is structured as follows. Section 3.2 summarizes the discontinuous Galerkin scheme that was presented in Ref. [1] and that we employ to discretize all elliptic equations in this article. Section 3.3 details the stack of task-based algorithms that constitutes the elliptic solver, and that we have implemented in the `SpECTRE` code. In Section 3.4 we assess the performance and parallel efficiency of our new elliptic solver by applying it to a set of test problems. We conclude in Section 3.5.

3.2 Discontinuous Galerkin discretization

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

We employ the discontinuous Galerkin (DG) scheme developed in Ref. [1] to discretize all elliptic problems in this article and summarize it in this section.

Schematically, the discretization procedure translates a linear elliptic problem to a matrix equation, such as

$$-\partial_i \partial_i \varphi(\boldsymbol{x}) = 4\pi\rho(\boldsymbol{x}) \quad \xrightarrow[\text{Ref. [1]}]{} \quad \mathcal{A}\underline{u} = \underline{b}, \tag{3.1}$$

where $\underline{u} = (u_1, \ldots, u_{N_{\text{DOF}}})$ is a discrete representation of all variables on the computational grid, $\underline{b} = (b_1, \ldots, b_{N_{\text{DOF}}})$ is a discrete representation of the fixed sources in the PDEs, and $\mathcal{A}$ is an $N_{\text{DOF}} \times N_{\text{DOF}}$ matrix that represents the discrete Laplacian operator in this example. Equation (3.1) represents the Maxwell constraint equation for the electric potential $\varphi(\boldsymbol{x})$ in Coulomb gauge, written here in Cartesian coordinates, where $\rho(\boldsymbol{x})$ is the electric charge density sourcing the field. We employ the Einstein sum convention to sum over repeated indices.

The subject of this section is to define the matrix equation (3.1) for a wide range of elliptic problems, as detailed in Ref. [1]. Then, the remainder of this article is concerned with solving the matrix equation numerically for $\underline{u}$, and doing so iteratively, in parallel on computing clusters, and without ever explicitly constructing the full matrix $\mathcal{A}$. Instead, we only need to define the matrix-vector product $\mathcal{A}\underline{u}$. We solve nonlinear problems $\mathcal{A}(\underline{u}) = \underline{b}$ by repeatedly solving their linearization.

Skip ahead

The remainder of this section summarizes the preceding Chapter 2. If you have read Chapter 2, skip ahead to Section 3.3.

The discontinuous Galerkin scheme detailed in Ref. [1] applies to a wide range of elliptic problems. Specifically, it applies to any set of elliptic PDEs that admits a formulation in first-order flux form

$$-\partial_i \mathcal{F}_\alpha^{\ i}[u_A, v_A; \boldsymbol{x}] + \mathcal{S}_\alpha[u_A, v_A; \boldsymbol{x}] = f_\alpha(\boldsymbol{x}), \tag{3.2}$$

where the fluxes $\mathcal{F}_\alpha{}^i$ and the sources $\mathcal{S}_\alpha$ are functionals of a set of *primal* variables $u_A(\boldsymbol{x})$ and *auxiliary* variables $v_A(\boldsymbol{x})$, and the fixed sources $f_\alpha(\boldsymbol{x})$ are functions of coordinates. The index α enumerates both primal and auxiliary equations. The primal variables can be scalars, such as the electric potential $\varphi(\boldsymbol{x})$ in the Maxwell constraint (3.1), higher-rank tensor fields such as the displacement vector in an elasticity problem, or combinations thereof such as in Eq. (3.4) below. The auxiliary variables are typically gradients of the primal variables, such as $v_i = \partial_i \varphi(\boldsymbol{x})$ for the Maxwell constraint. For example, the Maxwell constraint (3.1) can be formulated with the fluxes and sources

$$\mathcal{F}_{v\,j}{}^i = \varphi\,\delta^i_j, \quad \mathcal{S}_{v\,j} = v_j, \quad f_{v\,j} = 0, \tag{3.3a}$$

$$\mathcal{F}_\varphi{}^i = v_i, \qquad \mathcal{S}_\varphi = 0, \qquad f_\varphi = 4\pi\rho(\boldsymbol{x}), \tag{3.3b}$$

where δ^i_j denotes the Kronecker delta. Note that Eq. (3.3a) is the definition of the auxiliary variable, and Eq. (3.3b) is the Maxwell constraint (3.1).

In particular, the flux form (3.2) also encompasses the extended conformal thin-sandwich (XCTS) formulation of the Einstein constraint equations [74],[1]

$$\bar{\nabla}^2\psi = \frac{1}{8}\psi\bar{R} + \frac{1}{12}\psi^5 K^2 - \frac{1}{8}\psi^{-7}\bar{A}_{ij}\bar{A}^{ij} - 2\pi\psi^5\rho \tag{3.4a}$$

$$\bar{\nabla}^2(\alpha\psi) = \alpha\psi\left(\frac{7}{8}\psi^{-8}\bar{A}_{ij}\bar{A}^{ij} + \frac{5}{12}\psi^4 K^2 + \frac{1}{8}\bar{R} + 2\pi\psi^4(\rho + 2S)\right) - \psi^5\partial_t K + \psi^5\beta^i\bar{\nabla}_i K \tag{3.4b}$$

$$\bar{\nabla}_i(\bar{L}\beta)^{ij} = (\bar{L}\beta)^{ij}\bar{\nabla}_i\ln(\bar{\alpha}) + \bar{\alpha}\bar{\nabla}_i\left(\bar{\alpha}^{-1}\bar{u}^{ij}\right) + \frac{4}{3}\bar{\alpha}\psi^6\bar{\nabla}^j K + 16\pi\bar{\alpha}\psi^{10}S^j \tag{3.4c}$$

with $\bar{\nabla}^2 = \bar{\gamma}^{ij}\bar{\nabla}_i\bar{\nabla}_j$, $\bar{A}^{ij} = \frac{1}{2\bar{\alpha}}\left((\bar{L}\beta)^{ij} - \bar{u}^{ij}\right)$ and $\bar{\alpha} = \alpha\psi^{-6}$. The XCTS equations are a set of coupled nonlinear elliptic PDEs that the spacetime metric of general relativity must satisfy at all times. They are solved for the conformal factor ψ, the product of lapse and conformal factor $\alpha\psi$, and the shift vector β^j. The remaining quantities in the equations, i.e., the conformal metric $\bar{\gamma}_{ij}$, the trace of the extrinsic curvature K, their respective time derivatives $\bar{u}_{ij}$ and $\partial_t K$, the energy density ρ, the stress-energy trace S and the momentum density S^i, are freely-specifiable fields that define the scenario at hand. In particular, the conformal metric $\bar{\gamma}_{ij}$ defines the background geometry of the elliptic problem, which determines the covariant derivative $\bar{\nabla}$, the Ricci scalar $\bar{R}$ and the longitudinal operator

$$\left(\bar{L}\beta\right)^{ij} = \bar{\nabla}^i\beta^j + \bar{\nabla}^j\beta^i - \frac{2}{3}\bar{\gamma}^{ij}\bar{\nabla}_k\beta^k. \tag{3.5}$$

Reference [1] lists fluxes and sources for the XCTS equations, and for a selection of other elliptic systems.

Once we have formulated the equations, we choose a computational domain on which to discretize them. We decompose the d-dimensional computational domain $\Omega \subset \mathbb{R}^d$ into a set of *blocks* shaped like deformed cubes, as illustrated in Fig. 3.1a. Blocks do not overlap, but they share

[74]: Pfeiffer (2005), *The Initial value problem in numerical relativity*

1: See Section 1.2.3 for an introduction to the XCTS equations, and note that I use the symbol ∇_i instead of $\mathcal{D}_i$ for spatial covariant derivatives in this article because no spacetime covariant derivatives appear.

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

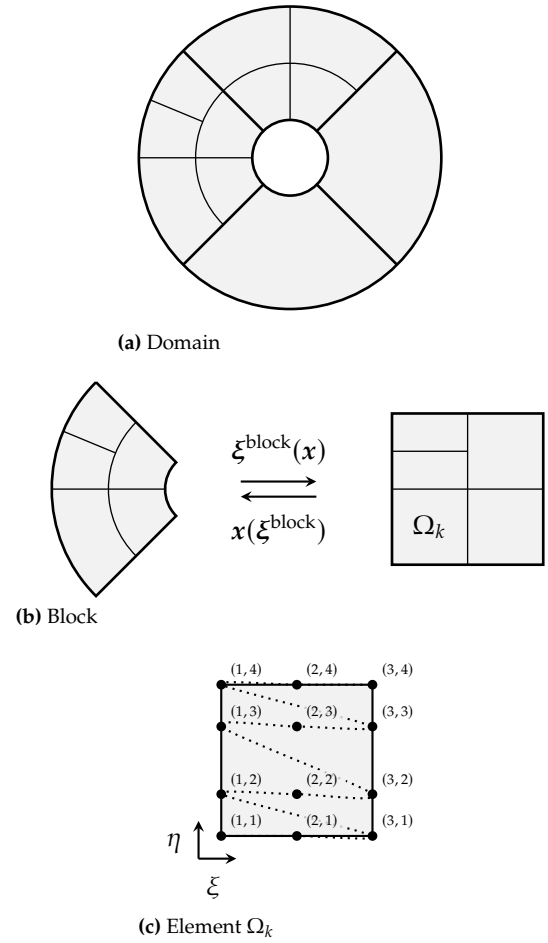


Figure 3.1: *Top:* Geometry of a two-dimensional computational domain composed of four wedge-shaped blocks. *Middle:* The coordinate transformation $\boldsymbol{\xi}^{\text{block}}(\boldsymbol{x})$ maps a block to a reference cube in block-logical coordinates $[-1,1]^2$. A block is split into elements Ω_k along its logical coordinates axes. *Bottom:* The element Ω_k in element-logical coordinates $\boldsymbol{\xi} = (\xi, \eta)$ with its grid of Legendre-Gauss-Lobatto collocation points. In this example we chose $N_{k,\xi} = 3$ and $N_{k,\eta} = 4$. Each grid point is labeled with its index (p_ξ, p_η). The dotted line connects points in the order they are enumerated in by the index p.

boundaries. Each face of a block is either shared with precisely one other block, or is external. For example, the domain depicted in Fig. 3.1a has four wedge-shaped blocks. Each block $B \subset \Omega$ carries a map from the coordinates $\boldsymbol{x} \in B$, in which the elliptic equations (3.2) are formulated, to *block-logical* coordinates $\boldsymbol{\xi}^{\text{block}} \in [-1,1]^d$ representing the d-dimensional reference cube, as illustrated in Fig. 3.1b.

Blocks decompose into *elements* $\Omega_k \subset \Omega$, by recursively splitting in half along any of their logical coordinate axes (h refinement). We limit the h refinement of our computational domain such that an element shares its boundary with at most two neighbors per dimension in every direction ("two-to-one balance"), both within a block and across block boundaries. Each element defines element-logical coordinates $\boldsymbol{\xi} \in [-1,1]^d$ by an affine transformation of the block-logical coordinates. The resulting coordinate map to the reference cube of the element is characterized by the Jacobian

$$\mathrm{J}^i_j = \frac{\partial x^i}{\partial \xi^j} \tag{3.6}$$

with determinant J and inverse $(\mathrm{J}^{-1})^j_i = \partial \xi^j / \partial x^i$.

On the reference cube of the element we choose a regular grid of collocation points along the logical coordinate axes, as illustrated in Fig. 3.1c (p refinement). Specifically, we choose $N_{k,i}$ Legendre-Gauss-Lobatto (LGL) collocation points, ξ_{p_i}, in each dimension i, where the index $p_i \in \{1, \ldots, N_{k,i}\}$ identifies the grid point along dimension i. We also enu-

merate all $N_k = \prod_i N_{k,i}$ d-dimensional grid points $\boldsymbol{\xi}_p = (\xi_{p_1}, \dots, \xi_{p_d})$ in an element with a single index $p \in \{1, \dots, N_k\}$, as illustrated in Fig. 3.1c.

Then, fields are represented numerically by their values at the collocation points. We denote the set of discrete field values within an element Ω_k as $\underline{u}^{(k)} = (u_1, \dots, u_{N_k})$, and the collection of discrete field values over *all* elements as $\underline{u}$. The field values at the collocation points within an element define a d-dimensional Lagrange interpolation,

$$u^{(k)}(\boldsymbol{x}) \coloneqq \sum_{p=1}^{N_k} u_p \psi_p(\boldsymbol{\xi}(\boldsymbol{x})) \quad \text{with} \quad \boldsymbol{x} \in \Omega_k, \tag{3.7}$$

where the basis functions $\psi_p(\boldsymbol{\xi})$ are products of Lagrange polynomials,

$$\psi_p(\boldsymbol{\xi}) \coloneqq \prod_{i=1}^{d} \ell_{p_i}(\xi^i) \quad \text{with} \quad \boldsymbol{\xi} \in [-1, 1]^d. \tag{3.8}$$

based on the collocation points in dimension i of the element. Since Eqs. (3.7) and (3.8) are local to each element, fields over the entire domain are discontinuous across element boundaries.

Finally, we employ the strong discontinuous Galerkin scheme developed in Ref. [1] to discretize the equations in first-order flux form, Eq. (3.2). To compute the matrix-vector product in Eq. (3.1) we first compute the auxiliary variables v_A, given the primal variables u_A, as

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

$$v_A = \mathrm{D}_i \cdot {\mathcal{F}_{v_A}}^i + \mathrm{L} \cdot \left((n_i {\mathcal{F}_{v_A}}^i)^* - n_i {\mathcal{F}_{v_A}}^i \right) - \tilde{\mathcal{S}}_{v_A}, \tag{3.9a}$$

where we assume the auxiliary sources can be written in the form $\mathcal{S}_{v_A} = v_A + \tilde{\mathcal{S}}_{v_A}[u_A; \boldsymbol{x}]$ such that Eq. (3.9a) depends only on the primal variables. We also assume $f_{v_A} = 0$ for convenience. All elliptic equations that we consider in this article fulfill these assumptions. In a second step, we use the computed auxiliary variables v_A, as well as the primal variables u_A, to compute the DG residuals

$$-\mathrm{MD}_i \cdot {\mathcal{F}_{u_A}}^i - \mathrm{ML} \cdot \left((n_i {\mathcal{F}_{u_A}}^i)^* - n_i {\mathcal{F}_{u_A}}^i \right) + \mathrm{M} \cdot \mathcal{S}_{u_A} = \mathrm{M} \cdot f_{u_A}. \tag{3.9b}$$

The operation $\cdot$ in Eq. (3.9) denotes a matrix multiplication with the field values over the computational grid of an element. We make use of the mass matrix

$$\mathrm{M}_{pq} = \int_{[-1,1]^d} \psi_p(\boldsymbol{\xi}) \psi_q(\boldsymbol{\xi}) \sqrt{g}\, \mathrm{J}\, \mathrm{d}^d \xi\,, \tag{3.10}$$

the stiffness matrix

$$\mathrm{MD}_{i,pq} = \int_{[-1,1]^d} \psi_p(\boldsymbol{\xi})\, \frac{\partial \psi_q}{\partial \xi^j}(\boldsymbol{\xi})\, (\mathrm{J}^{-1})^j_i \sqrt{g}\, \mathrm{J}\, \mathrm{d}^d \xi\,, \tag{3.11}$$

and the lifting operator

$$\mathrm{ML}_{pq} = \int_{[-1,1]^{d-1}} \psi_p(\boldsymbol{\xi}) \psi_q(\boldsymbol{\xi}) \sqrt{g^\Sigma}\, \mathrm{J}^\Sigma\, \mathrm{d}^{d-1} \xi \tag{3.12}$$

on the element Ω_k, as well as the associated "massless" operators $\mathrm{D}_i \coloneqq \mathrm{M}^{-1}\mathrm{MD}_i$ and $\mathrm{L} \coloneqq \mathrm{M}^{-1}\mathrm{ML}$. Here, $\sqrt{g}$ denotes the determinant

of the metric on which the elliptic equations are formulated, such as the conformal metric $\bar{\gamma}_{ij}$ in the XCTS equations (3.4). The integral in Eq. (3.12) is over the boundary of the element, $\partial\Omega_k$, where n_i is the outward-pointing unit normal one-form, g^Σ is the surface metric determinant induced by the background metric, and J^Σ is the determinant of the surface Jacobian.

The quantities $(n_i \mathcal{F}_{v_A}{}^i)^*$ and $(n_i \mathcal{F}_{u_A}{}^i)^*$ in Eq. (3.9) denote a numerical flux that couples grid points across nearest-neighbor element boundaries. We employ the generalized internal-penalty numerical flux developed in Ref. [1],

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

$$(n_i \mathcal{F}_{v_A}{}^i)^* = \frac{1}{2}\Big[n_i^{\mathrm{int}} \mathcal{F}_{v_A}{}^i(u_A^{\mathrm{int}}) - n_i^{\mathrm{ext}} \mathcal{F}_{v_A}{}^i(u_A^{\mathrm{ext}})\Big], \tag{3.13a}$$

$$\begin{aligned}(n_i \mathcal{F}_{u_A}{}^i)^* = \frac{1}{2}\Big[&n_i^{\mathrm{int}} \mathcal{F}_{u_A}{}^i\big(\partial_j \mathcal{F}_{v_A}{}^j(u_A^{\mathrm{int}}) - \tilde{\mathcal{S}}_{v_A}(u_A^{\mathrm{int}})\big) \\ &- n_i^{\mathrm{ext}} \mathcal{F}_{u_A}{}^i\big(\partial_j \mathcal{F}_{v_A}{}^j(u_A^{\mathrm{ext}}) - \tilde{\mathcal{S}}_{v_A}(u_A^{\mathrm{ext}})\big)\Big] \\ -\sigma\Big[&n_i^{\mathrm{int}} \mathcal{F}_{u_A}{}^i\big(n_j^{\mathrm{int}} \mathcal{F}_{v_A}{}^j(u_A^{\mathrm{int}})\big) \\ &- n_i^{\mathrm{ext}} \mathcal{F}_{u_A}{}^i\big(n_j^{\mathrm{ext}} \mathcal{F}_{v_A}{}^j(u_A^{\mathrm{ext}})\big)\Big]\end{aligned} \tag{3.13b}$$

with the penalty function

$$\sigma = C\,\frac{\big(\max(p^{\mathrm{int}}, p^{\mathrm{ext}}) + 1\big)^2}{\min(h^{\mathrm{int}}, h^{\mathrm{ext}})}. \tag{3.14}$$

Here, u_A^{int} denotes the primal variables on the *interior* side of an element's shared boundary with a neighbor, and u_A^{ext} denotes the primal variables on the neighbor's side, i.e., the *exterior*. Note that $n_i^{\mathrm{ext}} = -n_i^{\mathrm{int}}$ for the purpose of this article, since we only consider equations formulated on a fixed background metric, but the scheme does not rely on this assumption. For Eq. (3.14) we also make use of the polynomial degree p, and a measure of the element size, h, orthogonal to the element boundary on either side of the interface, as detailed in Ref. [1].

We impose boundary conditions through fluxes, i.e., by a choice of exterior quantities in the numerical flux, Eq. (3.13). Specifically, on external boundaries we set

$$(n_i \mathcal{F}_\alpha{}^i)^{\mathrm{ext}} = (n_i \mathcal{F}_\alpha{}^i)^{\mathrm{int}} - 2(n_i \mathcal{F}_\alpha{}^i)^{\mathrm{b}}, \tag{3.15}$$

where we choose the boundary fluxes $(n_i \mathcal{F}_\alpha{}^i)^{\mathrm{b}}$ depending on the boundary conditions we intend to impose. For *Neumann-type* boundary conditions we choose the primal boundary fluxes $(n_i \mathcal{F}_{u_A}{}^i)^{\mathrm{b}}$ directly, e.g., $(n_i \mathcal{F}_\varphi{}^i)^{\mathrm{b}} = n_i \partial_i \varphi|_{\mathrm{b}}$ for the Maxwell constraint (3.1), and set the auxiliary boundary fluxes to their interior values, $(n_i \mathcal{F}_{v_A}{}^i)^{\mathrm{b}} = (n_i \mathcal{F}_{v_A}{}^i)^{\mathrm{int}}$. For *Dirichlet-type* boundary conditions we choose the primal boundary fields u_A^{b}, e.g., $\varphi|_{\mathrm{b}}$ for the Maxwell constraint (3.1), to compute the auxiliary boundary fluxes $(n_i \mathcal{F}_{v_A}{}^i)^{\mathrm{b}} = n_i^{\mathrm{b}} \mathcal{F}_{v_A}{}^i(u_A^{\mathrm{b}})$, and set the primal boundary fluxes to their interior values, $(n_i \mathcal{F}_{u_A}{}^i)^{\mathrm{b}} = (n_i \mathcal{F}_{u_A}{}^i)^{\mathrm{int}}$.

In summary, the DG residuals (3.9) are algebraic equations for the discrete primal field values $\underline{u}_A$ on all elements and grid points in the computational domain. For linear PDEs, the left-hand side of Eq. (3.9b) defines a matrix-vector product with a set of primal field values on the

computational domain. The right-hand side of Eq. (3.9b) is a set of fixed values on the computational domain. Therefore, Eq. (3.9b) has the form of Eq. (3.1).

3.3 Task-based iterative algorithms

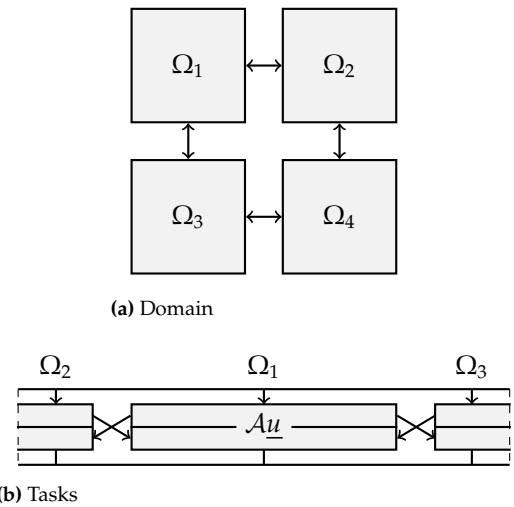
Once the elliptic problem is discretized, it is the responsibility of the elliptic solver to invert the matrix equation (3.1) numerically, in order to obtain the solution vector $\underline{u}$ over the computational grid. For large problems on high-resolution grids it is typically unfeasible to invert the matrix $\mathcal{A}$ in Eq. (3.1) directly, or even to explicitly construct and store it.[2] Instead, we employ iterative algorithms that require only the matrix-vector product $\mathcal{A}\underline{u}$ be defined, and that parallelize to computing clusters.

2: The KADATH [133] code explicitly constructs, stores and inverts the matrix $\mathcal{A}$, which it obtains by a spectral discretization, by distributing its columns over the available cores. Papenfort et al. [89] and Grandclément [133] quote the high memory demand of storing the explicit matrix and list iterative approaches to solve the linear system as a possible resolution. Such iterative approaches, and their parallelization, are the main focus of this article.
[89]: Papenfort et al. (2021), *New public code for initial data of unequal-mass, spinning compact-object binaries*
[133]: Grandclément (2010), *KADATH: A spectral solver for theoretical physics*

The discontinuous Galerkin (DG) matrix-vector product $\mathcal{A}\underline{u}$ is well suited for parallelization. As Section 3.2 summarized, it decomposes into a set of operations local to the elements that make up the computational domain. Figure 3.2 illustrates a computational domain composed of elements, as well as the dependence of the elements on each other for computing the matrix-vector product. The matrix-vector product requires only data local to each element and on both sides of the boundary that the element shares with its nearest neighbors. Therefore, it can be computed in parallel, and requires only a single communication between each pair of nearest-neighbor elements to exchange data on their shared boundary. The matrix-vector product acts as a "soft" global synchronization point, meaning that it requires *all* elements have sent data to their neighbors before *all* elements can proceed with the algorithm, but individual elements can already proceed once they receive data from their nearest neighbors.

The decomposition of the domain into elements also admits a strategy to distribute computation across the processors of the computer system. We distribute elements among the available cores in a way that, ideally, minimizes the number of internode communications and assigns an equal amount of work to each core. In this article we employ a Morton ("z-order") space-filling curve [194] to traverse the elements within a

[194]: Sagan (1994), *Space-Filling Curves*



(a) Domain

(b) Tasks

Figure 3.2: Parallelization structure of the matrix-vector product $\mathcal{A}\underline{u}$. *Top:* Decomposition of a two-dimensional rectangular domain into four elements. Arrows illustrate the dependence between nearest-neighbor elements to compute the matrix-vector product $\mathcal{A}\underline{u}$. *Bottom:* Tasks involved to compute the matrix-vector product $\mathcal{A}\underline{u}$. Each element performs a task that prepares and sends data to its neighbors (upper half of the rectangle), and another that receives data from its neighbors and performs the computation (lower half of the rectangle). The arrows between elements are the same as in the top panel.

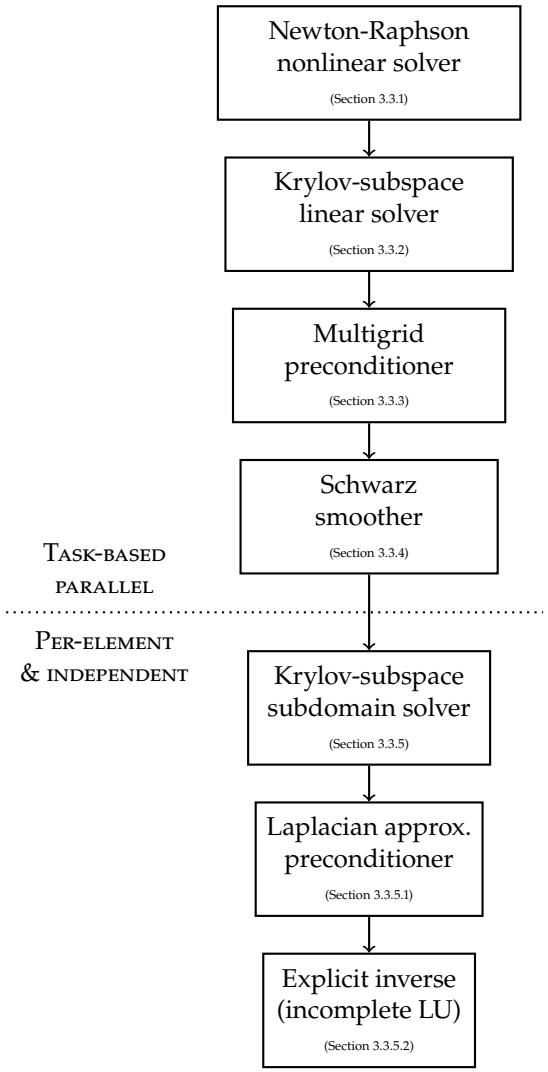


Figure 3.3: Overview of the technology stack we employ to solve the discretized elliptic problem (3.1). All algorithms above the dotted line follow `SpECTRE`'s task-based parallelism paradigm. The algorithms below the dotted line run within a task, and on all elements independently.

block of the computational domain and fill up the available cores. We weight the elements by their number of grid points to approximately balance the amount of work assigned to each core. With this strategy, neighboring elements tend to lie on the same node, though more effective element-distribution and load-balancing strategies based on, for example, Hilbert space-filling curves [195] are a subject of future work.

[195]: Borrell et al. (2018), *Parallel mesh partitioning based on space filling curves*

Once elements have been assigned to the available cores, each element traverses the list of tasks in the algorithm. When it encounters a task whose dependencies are not yet fulfilled, e.g., when neighbors have not yet sent the data on shared boundaries needed for the DG matrix-vector product, the element relinquishes control of the core to another whose dependencies are fulfilled. Reference [142] describes `SpECTRE`'s task-based parallel runtime system based on the `Charm++` [141] framework in more detail.

[142]: Kidder et al. (2017), *SpECTRE: A Task-based Discontinuous Galerkin Code for Relativistic Astrophysics*

[141]: The Charm++ Parallel Programming System, `https://charm.cs.illinois.edu`

Figure 3.3 provides an overview of the algorithms that we employ to iteratively solve the discretized elliptic problem (3.1), with details given in subsequent sections: nonlinear equations are linearized with a Newton-Raphson scheme with a line-search globalization (Section 3.3.1). The resulting linear subproblems are solved with an iterative Krylov-subspace method (Section 3.3.2), preconditioned with a multigrid solver

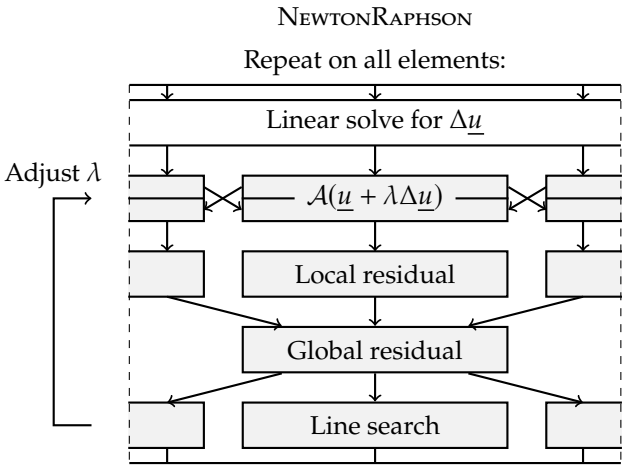


Figure 3.4: Parallelization structure of the task-based Newton-Raphson nonlinear solver (Section 3.3.1).

(Section 3.3.3). On every level of the multigrid hierarchy we run a few iterations of an additive Schwarz smoother, which solves the problem approximately on independent, overlapping, element-centered subdomains (Section 3.3.4). Each subdomain problem is solved by another Krylov-type method, which carries a Laplacian-approximation preconditioner with an incomplete LU explicit-inversion scheme to accelerate the solve (Section 3.3.5). All algorithms are implemented in the open-source `SpECTRE` code and take advantage of its task-based parallel infrastructure [10, 142].

[10]: SpECTRE, spectre-code.org
[142]: Kidder et al. (2017), *SpECTRE: A Task-based Discontinuous Galerkin Code for Relativistic Astrophysics*

3.3.1 Newton-Raphson nonlinear solver

The Newton-Raphson scheme iteratively refines an initial guess $\underline{u}_0$ for a nonlinear problem $\mathcal{A}(\underline{u}) = \underline{b}$ by repeatedly solving the linearized problem

$$\frac{\delta\mathcal{A}}{\delta u}(\underline{u})\,\Delta\underline{u} = \underline{b} - \mathcal{A}(\underline{u}) \tag{3.16}$$

for the correction $\Delta\underline{u}$, and then updating the solution as $\underline{u} \to \underline{u} + \Delta\underline{u}$ [124, 196].

[124]: Press et al. (2007), *Numerical Recipes*
[196]: Dennis and Schnabel (1996), *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*

The Newton-Raphson method converges quadratically once it reaches a basin of attraction, but can fail to converge when the initial guess is too far from the solution. We employ a line-search globalization strategy to recover convergence in such cases, following Alg. 6.1.3 in Dennis and Schnabel [196]. It iteratively reduces the step length λ until the corrected residual $\|\underline{b} - \mathcal{A}(\underline{u} + \lambda\Delta\underline{u})\|_2$ has sufficiently decreased, meaning it has decreased by a fraction of the predicted decrease if the problem was linear. This fraction is the *sufficient-decrease parameter* controlling the line search. The line search typically starts at $\lambda = 1$ in every Newton-Raphson iteration, but the initial step length can be decreased to dampen the nonlinear solver. Although the line-search globalization has proven effective for the cases we have encountered so far, alternative globalization strategies such as a trust-region method or more sophisticated nonlinear preconditioning techniques can be investigated in the future.[3]

3: See, e.g., Brune et al. [197] for an overview of nonlinear preconditioning techniques in the context of the `PETSc` [136] library.

[136]: PETSc, `https://www.mcs.anl.gov/petsc`
[197]: Brune et al. (2015), *Composing Scalable Nonlinear Algebraic Solvers*

Figure 3.4 illustrates our task-based implementation of the Newton-Raphson algorithm. The sufficient-decrease condition, and the necessity to check the global residual magnitude against convergence criteria,

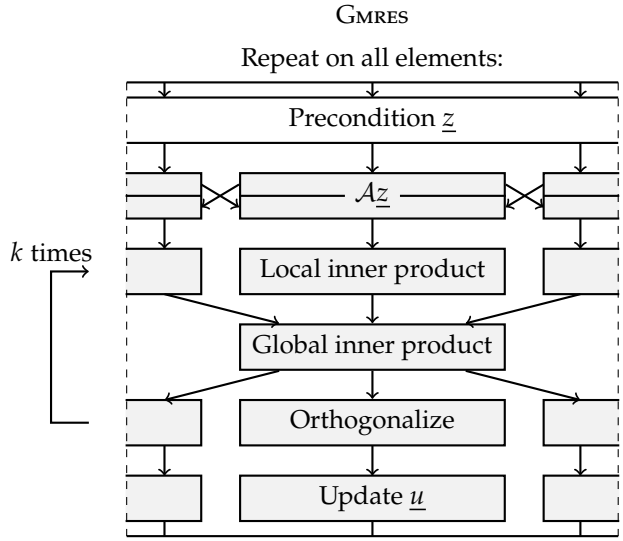


Figure 3.5: Parallelization structure of the task-based GMRES Krylov-subspace linear solver (Section 3.3.2).

introduce a synchronization point in the form of a global reduction to assemble the residual magnitude $\|\underline{b} - \mathcal{A}(\underline{u} + \lambda\Delta\underline{u})\|_2$. The algorithm requires one nonlinear operator application $\mathcal{A}(\underline{u} + \lambda\Delta\underline{u})$ per iteration, plus one additional nonlinear operator application for every globalization step that reduces the step length. Since a typical nonlinear elliptic solve requires $\lesssim 10$ Newton-Raphson iterations, the parallelization properties of this algorithm are not particularly important for the overall performance.

Exactly once per iteration the Newton-Raphson algorithm dispatches a linear solve of Eq. (3.16) for the correction $\Delta\underline{u}$. This iterative linear-solver algorithm is the subject of the following section.

3.3.2 Krylov-subspace linear solver

We solve the linearized problem (3.16) with an iterative Krylov-subspace algorithm. We generally employ a GMRES algorithm, but have also developed a conjugate gradients algorithm for discretized problems that are symmetric positive definite [127, 135]. These algorithms solve a linear problem $\mathcal{A}\underline{u} = \underline{b}$ iteratively by building up a basis of the Krylov subspace $\mathcal{K}_k = \text{span}\left\{\underline{b}, \mathcal{A}\underline{b}, \mathcal{A}^2\underline{b}, \ldots, \mathcal{A}^{k-1}\underline{b}\right\}$. Krylov-subspace algorithms are guaranteed to find a solution in at most N_{DOF} iterations, where N_{DOF} is the size of the matrix $\mathcal{A}$.

[127]: Saad (2003), *Iterative Methods for Sparse Linear Systems*
[135]: Saad and Schultz (1986), *GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems*

Figure 3.5 illustrates our task-based GMRES algorithm, which is based on Alg. 9.6 in Saad [127]. It requires one application of the linear operator $\mathcal{A}$ per iteration. Then, the algorithm is characterized by an Arnoldi orthogonalization procedure to construct a new basis vector $\underline{z}$ of the Krylov subspace that is orthogonal to all previously constructed basis vectors. The orthogonalization procedure requires a global reduction to assemble the inner product of the new basis vector with every existing basis vector, meaning the GMRES algorithm needs to perform k reductions in the kth iteration. Every reduction constitutes a global synchronization point, since it requires that all elements send data to a single core on the computer system and wait for a broadcast from that core back to all elements. A conjugate gradients algorithm also requires a

global reduction per iteration, but avoids the additional reductions from the orthogonalization procedure.

Due to the global synchronization points involved in every iteration of the Krylov-subspace solver, it is essential to keep the number of iterations to a minimum in order to achieve good parallel performance. To this end, we invoke a *preconditioner* in every iteration of the Krylov-subspace algorithm and place particular focus on its parallelization properties. The preconditioner is responsible for solving the linear problem approximately to accelerate the convergence of the Krylov-subspace algorithm.[4] Effective parallel preconditioning techniques for our DG-discretized elliptic problems are the main focus of this article. Since we employ the *flexible* variant of the GMRES algorithm, the preconditioner may change between iterations [127]. While the flexible GMRES algorithm with a variable preconditioner is not mathematically guaranteed to converge in at most N_{DOF} iterations anymore, in practice, it converges in much fewer than N_{DOF} iterations (see test problems in Section 3.4, in particular Figs. 3.12 and 3.15).[5]

4: See, e.g., Saad [127] for an introduction to iterative linear solvers and preconditioning techniques.

5: See also Sec. 9.4.1 in Saad [127] for a discussion of the flexible GMRES algorithm.

Typically, the number of iterations needed by an unpreconditioned Krylov-subspace algorithm increases with the size of the problem. The convergence behavior is often connected to the condition number of the linear operator,

$$\kappa = \frac{\lambda_{\max}}{\lambda_{\min}}, \tag{3.17}$$

where $\lambda_{\max}$ and $\lambda_{\min}$ denote the largest and smallest eigenvalue of the matrix, respectively. However, note that rigorous convergence bounds for the GMRES algorithm in terms of the condition number exist only when the matrix is *normal* [127]. Nevertheless, the condition number can provide an indication for the expected rate of convergence. For the discontinuous Galerkin discretization we employ in this article, the condition number scales as $\kappa \propto p^2/h$, where p denotes a typical polynomial degree of the elements and h denotes a typical element size [1, 8, 126, 162]. This scaling is related to the decrease of the minimum spacing between Legendre-Gauss-Lobatto collocation points, which scales quadratically with p near element boundaries and linearly with the element size.

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.
[8]: Vincent, Pfeiffer, and **Fischer** (2019), *hp-adaptive discontinuous Galerkin solver for elliptic equations in numerical relativity*
[126]: Hesthaven and Warburton (2008), *Nodal Discontinuous Galerkin Methods*
[162]: Shahbazi (2005), *An explicit expression for the penalty parameter of the interior penalty method*

More specifically, Krylov-subspace methods struggle to solve large-scale modes in the solution. The algorithm solves modes on the scale of the grid-point spacing or the size of elements in just a few iterations, but it needs a lot more iterations to solve modes spanning the full domain. Such large-scale modes carry, for example, information from boundary conditions that must traverse the entire domain. The test problem presented in Fig. 3.11 below illustrates this effect. Therefore, we precondition the Krylov-subspace solver with a multigrid algorithm that uses information from coarser grids, where the large-scale modes from finer grids become small scale.

3.3.3 Multigrid preconditioner

[8]: Vincent, Pfeiffer, and **Fischer** (2019), *hp-adaptive discontinuous Galerkin solver for elliptic equations in numerical relativity*

We employ a geometric V-cycle multigrid algorithm, as prototyped in Ref. [8].[6] Our multigrid solver can be used standalone, or to precondition a Krylov-type linear solver as described in Section 3.3.2 ("Krylov-accelerated multigrid").

6: See, e.g., Briggs, Henson, and McCormick [198] for an introduction to multigrid methods.

[198]: Briggs, Henson, and McCormick (2000), *A Multigrid Tutorial*

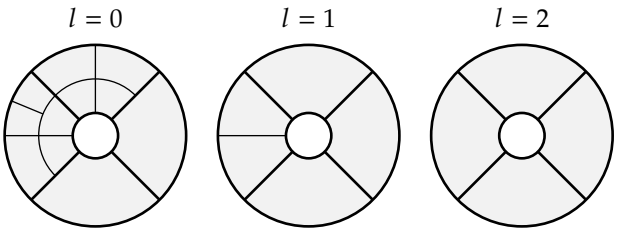


Figure 3.6: Multigrid hierarchy based on the domain depicted in Fig. 3.1a.

Grid hierarchy

The geometric multigrid algorithm relies on a strategy to coarsen the computational grid. We primarily h-coarsen the domain, meaning that we create multigrid levels $l > 0$ by successively combining two elements into one along every dimension of the grid, as illustrated in Fig. 3.6. We only p-coarsen the grid in the sense that we choose the smaller of the two polynomial degrees when combining elements along an axis. This strategy follows Ref. [8] and ensures that coarse-grid field approximations always have an exact polynomial representation on finer grids.

[8]: Vincent, Pfeiffer, and **Fischer** (2019), *hp-adaptive discontinuous Galerkin solver for elliptic equations in numerical relativity*

The coarsest possible grid that our domain decomposition can achieve has a single element per block that make up the domain. For example, the two-dimensional shell depicted in Fig. 3.6 has four wedge-shaped blocks, each of which is a deformed cube. Our multigrid algorithm works best when the domain is composed of as few blocks as possible.

Intermesh operators

To project data between grids we use the standard L_2-projections (or *Galerkin* projections) detailed in Ref. [1].[7] Fields on coarser grids are projected to finer grids with the *prolongation operator*

$$\mathrm{P}^{l+1\to l}_{\tilde{p}p} = \prod_{i=1}^{d} \ell_{p_i}(\tilde{\xi}_{\tilde{p}_i}), \tag{3.18}$$

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

7: See also Fortunato, Rycroft, and Saye [158] for details on the intermesh operators.

[158]: Fortunato, Rycroft, and Saye (2019), *Efficient Operator-Coarsening Multigrid Schemes for Local Discontinuous Galerkin Methods*

where p enumerates grid points on the coarser grid, $\tilde{p}$ enumerates grid points on the finer grid, and $\tilde{\xi}_{\tilde{p}_i}$ are the coarse-grid logical coordinates of the fine-grid collocation points. For fine-grid (child) elements that cover the full coarse-grid (parent) element in dimension i the coarse-grid logical coordinates are just the fine-grid collocation points, $\tilde{\xi}_{\tilde{p}_i} = \xi_{\tilde{p}_i}$. For child elements that cover the lower or upper logical half of the parent element in dimension i they are $\tilde{\xi}_{\tilde{p}_i} = (\xi_{\tilde{p}_i} - 1)/2$ or $\tilde{\xi}_{\tilde{p}_i} = (\xi_{\tilde{p}_i} + 1)/2$, respectively. Note that the prolongation operator (3.18) is just a Lagrange interpolation from the coarser to the finer grid. The interpolation retains the accuracy of the polynomial approximation because the finer grid always has sufficient resolution.

To project data from finer to coarser grids we employ the *restriction operator*

$$\mathrm{R}^{l\to l+1} = (\mathrm{P}^{l+1\to l})^T, \tag{3.19}$$

which is the transpose of the prolongation operator (3.18). Contrary to the restriction operator listed in Ref. [1] the multigrid restriction involves no mass matrices because it applies to DG residuals, Eq. (3.9b), which already include mass matrices.

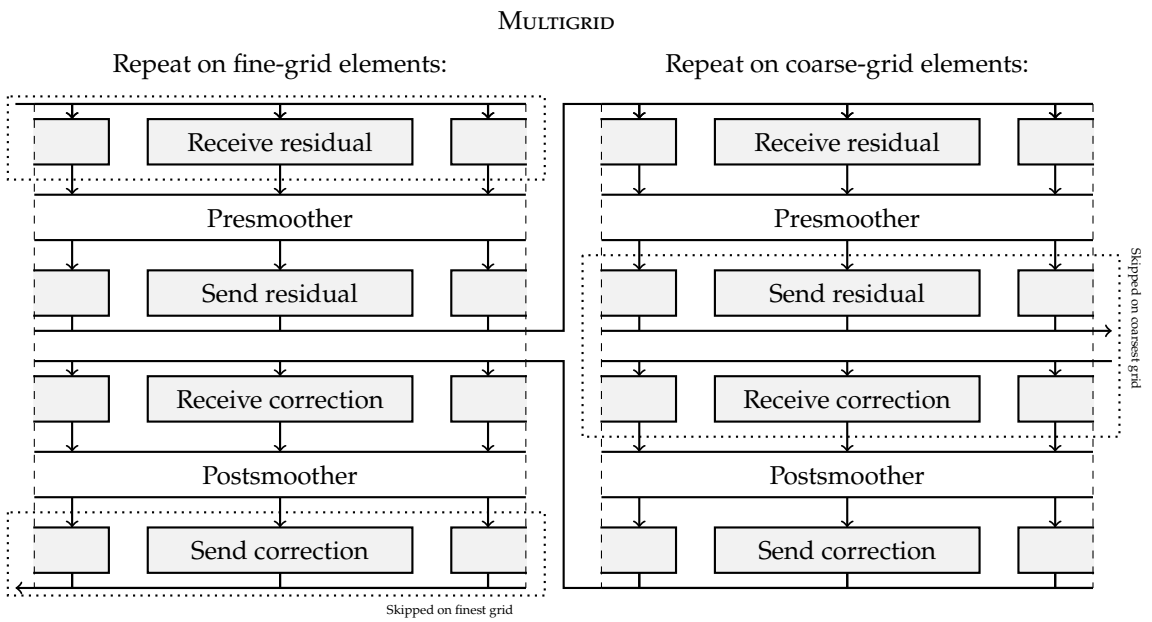


Figure 3.7: Parallelization structure of the task-based multigrid algorithm (Section 3.3.3). Elements on all grids perform the same set of tasks, with some tasks skipped on the finest grid and other tasks skipped on the coarsest grid.

Algorithm

Figure 3.7 illustrates our task-based implementation of the multigrid V-cycle algorithm to solve linear problems $\mathcal{A}\underline{u} = \underline{b}$. On every grid l we approximately solve the linear problem

$$\mathcal{A}_l \, \underline{u}^{(l)} = \underline{b}^{(l)}, \tag{3.20}$$

where the operator $\mathcal{A}_l$ is the discretization of the elliptic PDEs on the grid l. At the beginning of a V-cycle, on the finest grid $l = 0$, we select $\underline{u}^{(0)} = \underline{u}$ and $\underline{b}^{(0)} = \underline{b}$; hence, approximately solving the original linear problem ("presmoothing"). Then, the remaining residual

$$\underline{r}^{(l)} = \underline{b}^{(l)} - \mathcal{A}_l \, \underline{u}^{(l)} \tag{3.21}$$

is restricted to source the linear problem (3.20) on the next-coarser grid,

$$\underline{b}^{(l+1)} = \mathrm{R}^{l \to l+1} \, \underline{r}^{(l)}. \tag{3.22}$$

Once presmoothing is complete on the coarsest grid (the "tip" of the V-cycle), we approximately solve Eq. (3.20) again ("postsmoothing"). The solution of the postsmoothing step is prolongated to the next-finer grid as a correction,

$$\underline{u}^{(l)} \quad \leftarrow \quad \underline{u}^{(l)} + \mathrm{P}^{l+1 \to l} \, \underline{u}^{(l+1)}. \tag{3.23}$$

Prolongation, correction, and postsmoothing proceed until we have returned to the finest grid, where the correction and postsmoothing apply to the original linear problem. Our choice of presmoother and postsmoother to approximately solve Eq. (3.20) is detailed in Section 3.3.4 below. Note that on the coarsest level we apply both presmoothing and postsmoothing.

The restriction of residuals to the next-coarser grid and the prolongation

of corrections to the next-finer grid incur soft synchronization points. Specifically, only once *all* elements on the finer grid have restricted their residuals to the coarser grid can *all* elements on the coarser grid proceed, though individual coarse-grid (parent) elements can already proceed once only their corresponding fine-grid (child) elements have sent the restricted residuals. The same applies to the prolongation of corrections from the parent elements back to their children. Therefore, parent elements on coarser grids ideally follow their children when distributed among the available cores, initially and at load-balancing operations.

In contrast to Krylov-subspace algorithms, the multigrid algorithm involves no global synchronization points. In particular, for diagnostic output we perform a reduction to compute the global residual norm $\|\underline{r}^{(l)}\|$ on every grid, but do so asynchronously in order to avoid a synchronization that is not algorithmically necessary. Therefore, we do not use the global residual norm as a convergence criterion for the multigrid solver. Instead, we run a fixed number of multigrid V-cycles, and typically only a single one to precondition a Krylov-subspace solver.

3.3.4 Schwarz smoother

On every level of the grid hierarchy the multigrid solver relies on a *smoother* that approximately solves Eq. (3.20). In principle, the smoother can be any linear solver, including a Krylov-subspace solver as detailed in Section 3.3.2. However, to achieve good parallel performance we have developed a highly asynchronous additive Schwarz smoother [8, 199, 200], that we employ for presmoothing and postsmoothing on every level of the multigrid hierarchy. Note that we also apply it on the coarsest grid, where some authors apply a dedicated *bottom smoother* instead [201, 202]. Since our coarsest grids are rarely reduced to a single element (see Section 3.3.3), we have, so far, preferred the asynchronous Schwarz smoother over direct bottom smoothers. Our Schwarz smoother can be used standalone, as a preconditioner for a Krylov-type linear solver, or as a smoother for the multigrid solver (which may, in turn, precondition a Krylov-type solver).

[8]: Vincent, Pfeiffer, and **Fischer** (2019), *hp-adaptive discontinuous Galerkin solver for elliptic equations in numerical relativity*
[199]: Lottes and Fischer (2005), *Hybrid Multigrid/Schwarz Algorithms for the Spectral Element Method*
[200]: Stiller (2016), *Robust multigrid for high-order discontinuous Galerkin methods: A fast Poisson solver suitable for high-aspect ratio Cartesian grids*
[201]: AlOnazi, Markomanolis, and Keyes (2017), *Asynchronous Task-Based Parallelization of Algebraic Multigrid*
[202]: Kang (2015), *Scalable implementation of the parallel multigrid method on massively parallel computers*

The additive Schwarz method works by solving many subproblems in parallel and combining their solutions as a weighted sum to converge towards the global solution. The decomposition into independent subproblems makes this linear solver very parallelizable. The Schwarz solver is based on our prototype in Ref. [8], with variations to the subdomain geometry and weighting to take better advantage of our task-based parallelism, and with novel subdomain preconditioning techniques.

Subdomain geometry

We partition the computational domain into overlapping, element-centered subdomains $S_k \subset \Omega$, which have a one-to-one association with the DG elements Ω_k. Each subdomain S_k is centered on the DG element Ω_k. It extends by N_{overlap} collocation points into neighboring elements across every face of Ω_k, up to, but excluding, the collocation

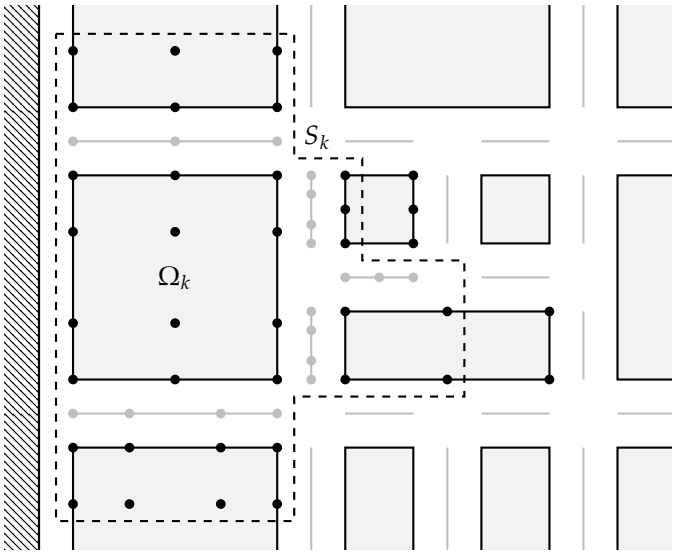


Figure 3.8: An element-centered subdomain S_k with $N_{\text{overlap}} = 2$ (dashed line) associated with the element Ω_k in a two-dimensional computational domain. The domain is composed of elements (black rectangles) with their mesh of grid points (black dots) and depicted here in block-logical coordinates. The diagonally-shaded region to the left illustrates an external domain boundary. The light gray lines between neighboring element faces illustrate mortar meshes, which are relevant for the subdomain operator in a DG context but play no role in the Schwarz algorithm [1]. Note that the empty space between the elements in this visualization is not part of the computational domain.

points on the face of the neighbor pointing away from the subdomain. Fig. 3.8 illustrates the geometry of our element-centered subdomains.

The subdomain does not extend into corner or edge neighbors, which is a choice different to both Ref. [200] and Ref. [8]. We avoid diagonal couplings because in a DG context information only propagates across faces, as already noted in Ref. [200]. Elimination of the corner and edge neighbors reduces the complexity of the subdomain geometry, the number of communications necessary to exchange data between elements in the subdomain, and hence the connectivity of the dependency graph between tasks. This element-centered subdomain geometry based solely on face neighbors has proven viable for the test problems presented below, and for our task-based parallel architecture.

[200]: Stiller (2016), *Robust multigrid for high-order discontinuous Galerkin methods: A fast Poisson solver suitable for high-aspect ratio Cartesian grids*

[8]: Vincent, Pfeiffer, and **Fischer** (2019), *hp-adaptive discontinuous Galerkin solver for elliptic equations in numerical relativity*

The one-to-one association between elements and subdomains allows to store all quantities that define the subdomain geometry local to the central element, i.e., on the same core. The same applies to all data on the grid points of the subdomain. Therefore, operations local to the subdomain require no communication, but communication between overlapping elements is necessary to assemble data on the subdomains, and to make data on subdomains available to overlapped elements.

Subdomain restriction

To restrict quantities defined on the full computational domain Ω to a subdomain $S \subset \Omega$ the Schwarz solver employs a *restriction operator* R_S. Since our subdomains are subsets of the grid points in the full computational domain, our restriction operator simply discards all nodal data on grid points outside the subdomain. Similarly, the transpose of the restriction operator, R_S^T, extends subdomain data with zeros on all grid points outside the subdomain.[8]

8: See also Sec. 3.1 in Stiller [200] for details on the subdomain restriction operation.

[200]: Stiller (2016), *Robust multigrid for high-order discontinuous Galerkin methods: A fast Poisson solver suitable for high-aspect ratio Cartesian grids*

The Schwarz solver also relies on a restriction of the global linear operator $\mathcal{A}$ to the subdomains. The subdomain operator $\mathcal{A}_S$ on a subdomain S is formally defined as $\mathcal{A}_S = \mathrm{R}_S \mathcal{A} \mathrm{R}_S^T$. In practice, it evaluates the same DG matrix-vector product as the full operator $\mathcal{A}$, i.e., the left-hand side of Eq. (3.9b), but assumes that all data outside the subdomain is zero.

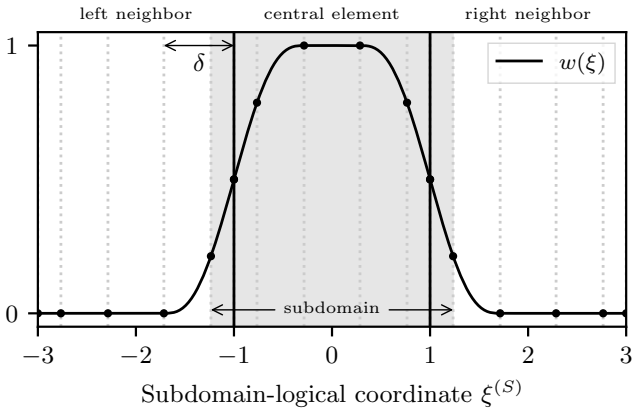


Figure 3.9: The one-dimensional weight function $w(\xi)$ for the Schwarz solver. Depicted is an element-centered subdomain in one dimension with $N_{\text{overlap}} = 2$. Every element has $N_k = 6$ LGL collocation points, which includes grid points on the shared element boundaries (black vertical lines). The overlap width δ is the logical coordinate distance to the first point outside the subdomain, where the weight becomes zero.

It performs all interelement operations of the full DG operator, but computes them entirely with data local to the subdomain. Therefore, it requires no communication between cores, as opposed to the global linear operator $\mathcal{A}$ that must communicate data between nearest neighbors for every operator application.

Subdomain problems

On every subdomain S we solve the restricted problem

$$\mathcal{A}_S \, \Delta \underline{u}^{(S)} = \underline{r}^{(S)} \tag{3.24}$$

for the subdomain correction $\Delta \underline{u}^{(S)}$. Here, $\mathcal{A}_S$ is the subdomain operator and $\underline{r}^{(S)} = \mathrm{R}_S \, \underline{r}$ is the global residual $\underline{r} = \underline{b} - \mathcal{A}\underline{u}$ restricted to the subdomain.

The subdomain problems (3.24) are solved by means of a *subdomain solver*, detailed in Section 3.3.5. The choice of subdomain solver affects only the performance of the Schwarz algorithm, not its convergence or parallelization properties, assuming the solutions to the subdomain problems (3.24) are sufficiently precise.

Weighting

Once we have obtained the subdomain correction $\Delta \underline{u}^{(S)}$ on every subdomain S, we combine them as a weighted sum to correct the solution,

$$\underline{u} \quad \leftarrow \quad \underline{u} + \sum_S \mathrm{R}_S^T \left(\underline{w}^{(S)} \, \Delta \underline{u}^{(S)} \right), \tag{3.25}$$

where $\underline{w}^{(S)}$ is a weight at every grid point of the subdomain. This is the *additive* approach of the algorithm, which has the advantage over *multiplicative* Schwarz methods that all subdomain problems decouple and can be solved in parallel.[9] The weighted sum, Eq. (3.25), is never assembled globally. Instead, every element adds the contribution from its locally centered subdomain and from all overlapping subdomains to the components of $\underline{u}$ that resides on the element.

9: See also Sec. 3.1 in Stiller [200] for details on multiplicative variants of Schwarz algorithms.

[200]: Stiller (2016), *Robust multigrid for high-order discontinuous Galerkin methods: A fast Poisson solver suitable for high-aspect ratio Cartesian grids*

The weights $\underline{w}^{(S)}$ represent a scalar field on every subdomain, which must be conserved as

$$\sum_S \mathrm{R}_S^T \, \underline{w}^{(S)} = \underline{1}. \tag{3.26}$$

We follow Refs. [8, 200] in constructing the weights as quintic smoothstep polynomials, but must account for the missing weight from corner and edge neighbors. Specifically, we compute

$$w_p^{(S)} = W(\boldsymbol{\xi}_p^{(S)}) \tag{3.27}$$

by evaluating the scalar weight function $W(\boldsymbol{\xi})$ at the logical coordinates $\boldsymbol{\xi}_p^{(S)}$ of the grid points in the subdomain. These subdomain-logical coordinates coincide with the element-logical coordinates of the central element, and extend outside the central element such that $\xi^{(S)} = \pm 3$ coincides with the sides of the overlapped neighbors that face away from the subdomain (see abscissa of Fig. 3.9). The scalar weight function

$$W(\boldsymbol{\xi}) = \prod_{i=0}^{d} w(\xi^i) \tag{3.28}$$

is a product of one-dimensional weight functions,

$$w(\xi) = \frac{1}{2}\left(\phi\left(\frac{\xi+1}{\delta}\right) - \phi\left(\frac{\xi-1}{\delta}\right)\right) \tag{3.29}$$

$$\text{with} \quad \phi(\xi) = \begin{cases} \frac{1}{8}\left(15\xi - 10\xi^3 + 3\xi^5\right) & \xi \in [-1,1] \\ \mathrm{sign}(\xi) & |\xi| > 1, \end{cases} \tag{3.30}$$

where $\phi(\xi)$ is a second-order smoothstep function, i.e., a quintic polynomial, and $\delta \in (0, 2]$ is the *overlap width*. The overlap width is the logical coordinate distance from the boundary of the central element to the first collocation point *outside* the overlap region (see Fig. 3.9). With this definition the overlap width is nonzero even when the overlap extends only to a single LGL point in the neighbor, which coincides with the element boundary. Furthermore, the weight is always zero at subdomain-logical coordinates $\xi^{(S)} = \pm 3$, even for $\delta = 2$ when the overlap region covers the full neighbor in width. This is the reason we never include the collocation points on the side of the neighbor facing away from the subdomain (see Section 3.3.4). Figure 3.9 illustrates the shape of the weight function.

To account for the missing weight from corner and edge neighbors, we could add it to the central element, to the overlap data from face neighbors, or split it between the two. We choose to add it to the face neighbors that share a corner or edge, since in a DG context that is where the information from those regions propagates through.

Algorithm

Fig. 3.10 illustrates our task-based implementation of the additive Schwarz solver. In each iteration, the algorithm computes the residual $\underline{r} = \underline{b} - \mathcal{A}\underline{u}$, restricts it to all subdomains as $\underline{r}^{(S)} = \mathrm{R}_S \, \underline{r}$, and exchanges it on overlap regions with neighboring elements. Once an element has received all residual data on its subdomain, it solves the subdomain problem, Eq. (3.24), for the correction $\Delta\underline{u}^{(S)}$. Since all elements perform such

[8]: Vincent, Pfeiffer, and **Fischer** (2019), *hp-adaptive discontinuous Galerkin solver for elliptic equations in numerical relativity*
[200]: Stiller (2016), *Robust multigrid for high-order discontinuous Galerkin methods: A fast Poisson solver suitable for high-aspect ratio Cartesian grids*

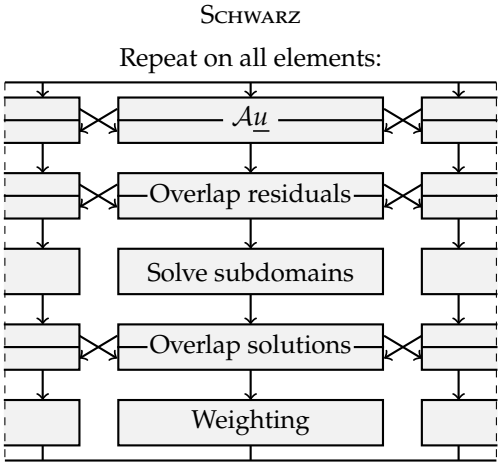


Figure 3.10: Parallelization structure of the task-based Schwarz smoother (Section 3.3.4).

a subdomain solve, we end up with a subdomain solution $\Delta\underline{u}^{(S)}$ on every element-centered subdomain, and the solutions overlap. Therefore, the algorithm exchanges the subdomain solutions on overlap regions with neighboring elements and adds them to the solution field $\underline{u}$ as the weighted sum, Eq. (3.25).

In order to compute the residual $\underline{r}$ that is restricted to the subdomains to serve as source for the subdomain solves, we must apply the global linear operator $\mathcal{A}$ to the solution field $\underline{u}$ once per Schwarz iteration. This operator application, as well as the steps to communicate the residuals and the solutions on overlaps, incur soft synchronization points through nearest-neighbor couplings. However, once the residuals on overlaps are communicated, all subdomain solves are independent of each other. This constitutes the main source of parallelization in the elliptic solver.

The subdomain solves not only run in parallel, but also scale with the problem size. Increasing the number of grid points in the elements (p refinement) makes the subdomain solves more expensive, but the effectiveness of a Schwarz iteration ideally remains the same. Increasing the number of elements (h refinement) leads to more subdomain solves that can run in parallel. The Schwarz solver does not resolve large-scale modes, so Krylov-type solvers still rely on the multigrid algorithm to scale with h refinement.

3.3.5 Subdomain solver

Once overlapping subdomains have exchanged data we can solve all subdomain problems (3.24) in parallel with data local to each subdomain. Since the subdomain operator $\mathcal{A}_S$ is defined as a matrix-vector product, we solve Eq. (3.24) with a preconditioned GMRES algorithm, or with conjugate gradients for symmetric positive definite problems. The algorithm is the same as detailed in Section 3.3.2 for our task-based parallel Krylov-subspace linear solver, but implemented separately as a serial algorithm. In future work, we may opt to parallelize the subdomain solver over a few threads with shared memory, but currently we prefer to employ the available cores to solve multiple subdomain problems in parallel. In particular, on coarse multigrid levels where the number

of elements can be smaller than the number of available cores, parallelizing the subdomain solves over otherwise idle cores may increase performance.

The iterative Krylov subdomain solves constitute the majority of the total computational expenses, so a suitable preconditioner for them can speed up the elliptic solve significantly. To our knowledge, preconditioners for Schwarz subdomain solvers have gotten little attention in the literature so far. In some cases, the discretization scheme allows to construct a matrix representation for the subdomain operator explicitly, making it possible to invert it directly with little effort [200]. In other cases, the subdomain operator is small enough to build the matrix representation column-by-column (see section on the explicit-inverse subdomain solver below), e.g., when solving the Poisson equation. However, when solving sets of coupled elliptic equations the subdomain operator can easily become too large to construct explicitly. For example, the subdomain operator for the XCTS equations (3.4) (five variables) on a three-dimensional grid with 8^3 grid points per element and $N_{\text{overlap}} = 2$ is a matrix of size 6400×6400. Stored densely, it requires over 300 MB of memory per element, so typical contemporary computing clusters with a few GB of memory per core could only hold a few elements per core. Sparse storage reduces the memory cost significantly, but still requires 6400 subdomain-operator applications to construct the matrix representation and a nonnegligible cost to invert and to apply it. With an iterative Krylov-subspace algorithm and a suitable preconditioner we can solve the subdomain problems on an element with significantly lower cost and memory requirements. For example, test problem 3.4.2 completes about an order of magnitude faster with the subdomain preconditioner laid out in this section, than with an unpreconditioned GMRES subdomain solver.

[200]: Stiller (2016), *Robust multigrid for high-order discontinuous Galerkin methods: A fast Poisson solver suitable for high-aspect ratio Cartesian grids*

Laplacian-approximation preconditioner

We support the iterative subdomain solver with a Laplacian-approximation preconditioner. It approximates the linearized elliptic PDEs with a Poisson equation for every variable. A similar preconditioning strategy has proven successful for the `SpEC` code [139], but in the context of a spectral discretization scheme and a very different linear-solver stack. Specifically, we approximate the subdomain problem, Eq. (3.24), as a set of independent Poisson subdomain problems

[139]: Pfeiffer et al. (2003), *A multidomain spectral method for solving elliptic equations*

$$\mathcal{A}_S^{\text{Poisson}} \, \Delta \underline{u}_A^{(S)} = \underline{r}_A^{(S)}, \tag{3.31}$$

where the index A iterates over all primal variables (see Section 3.2). Here, $\mathcal{A}_S^{\text{Poisson}}$ is the DG-discretization of the negative Laplacian $-g^{ij}\nabla_i\nabla_j$ on the subdomain according to Section 3.2. For example, a three-dimensional XCTS problem has five variables, so Eq. (3.31) approximates the linearization (3.16) of the five equations (3.4) as

$$\bar{\nabla}^2 \delta\psi = 0, \quad \bar{\nabla}^2 \delta(\alpha\psi) = 0 \quad \text{and} \quad \bar{\nabla}^2 \delta\beta^i = 0. \tag{3.32}$$

Depending on the elliptic system at hand we either choose a flat background metric $g_{ij} = \delta_{ij}$, or the background metric of the elliptic system, such as the conformal metric $g_{ij} = \bar{\gamma}_{ij}$ for an XCTS system. A curved

background metric reduces the sparsity of the Poisson operator but approximates the elliptic equations better. In practice, we have found little difference in runtime between the flat-space and curved-space Laplacian approximations.

We choose homogeneous Dirichlet or Neumann boundary conditions for $\mathcal{A}_S^{\text{Poisson}}$. For variables and element faces where the original boundary conditions are of Dirichlet type we choose homogeneous Dirichlet boundary conditions, and for those where the original boundary conditions are of Neumann type we choose homogeneous Neumann boundary conditions. This may lead to more than one distinct Poisson operator on subdomains with external boundaries, one per unique combination of element face and boundary-condition type among the variables. Subdomains that have exclusively internal boundaries only ever have a single Poisson operator, which applies to all variables. Note that the choice of homogeneous boundary conditions for the Poisson subdomain problems is compatible with inhomogeneous boundary value problems, because the inhomogeneity in the boundary conditions is absorbed in the fixed sources when the equations are linearized [1].

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

To solve the Poisson subdomain problems (3.31), one per variable, we can (again) employ any choice of linear solver, such as a (preconditioned) Krylov-subspace algorithm.[10] However, at this point we have reduced the full elliptic problem down to a single Poisson problem limited to a subdomain that is solved for all variables, or a few Poisson problems on subdomains with external boundaries. Therefore, it becomes feasible, and indeed worthwhile, to construct the Poisson subdomain-operator matrix explicitly and to invert it directly. In particular, the approximate Poisson subdomain-operator matrix remains valid throughout the full nonlinear elliptic solve, as long as the grid, the background metric, and the type of boundary conditions remain unchanged, so that its construction cost is amortized over many applications.

10: The absurdity of adding a *third* layer of nested preconditioned linear solvers was not lost on the authors.

Explicit-inverse solver

We solve the Poisson subproblems of the Laplacian-approximation preconditioner, Eq. (3.31), with an explicit-inverse solver. It constructs the matrix representation of a linear subdomain operator $\mathcal{A}_S$ column-by-column, and then inverts it directly by means of an LU decomposition. Once the inverse $\mathcal{A}_S^{-1}$ has been constructed, each subdomain problem $\mathcal{A}_S\,\Delta\underline{u}^{(S)} = \underline{r}^{(S)}$ is solved by a single application of the inverse matrix,

$$\Delta\underline{u}^{(S)} = \mathcal{A}_S^{-1}\,\underline{r}^{(S)}. \tag{3.33}$$

This means that subdomains have a large initialization cost, but fast repeated solves.

When the explicit-inverse solver is employed as a preconditioner, e.g., to solve the individual Poisson problems of the Laplacian-approximation preconditioner (Section 3.3.5), the inverse does not need to be exact. Therefore, we construct an *incomplete* LU decomposition with a configurable fill-in and store it in sparse format. Then, each subdomain problem reduces to two sparse triangular matrix solves. We use the `Eigen` [203] sparse linear algebra library for the incomplete LU decomposition, which

[203]: Eigen, `http://eigen.tuxfamily.org`

LUT: A dual threshold zation

uses the `ILUT` algorithm [204]. The Poisson subdomain-operator matrices $\mathcal{A}_S^{\text{Poisson}}$ have a sparsity of about 90 %, which translates to a sparsity of about 90 % for the incomplete LU decomposition as well, since we use a fill-in factor of one. The sparsity of the inverse reduces the computational cost for applying it to every subdomain problem, as well as the memory required to store the inverse.

Note that the explicit matrix must be reconstructed when the linear operator changes. The Poisson operators of the Laplacian-approximation preconditioner do not typically change, which makes the explicit-inverse solver very effective (see Section 3.3.5). However, in case we apply the explicit-inverse solver to the full subdomain problem directly, the linearized operator typically changes between every outer nonlinear solver iteration. In such cases, we can choose to skip the reconstruction of the explicit matrix to avoid the computational expense, at the cost of losing accuracy of the solver. When the reconstruction is skipped, the cached matrix only approximates the subdomain operator, but can still provide effective preconditioning.

3.4 Test problems

The following numerical tests demonstrate the accuracy, scalability, and parallel efficiency of the elliptic solver on a variety of linear and nonlinear elliptic problems.

All computations were performed on our local computing cluster `Minerva`. It is composed of 16-core nodes, each with two eight-core Intel Haswell E5-2630v3 processors clocked at 2.40 GHz and 64 GB of memory, connected with an Intel Omni-Path network. We distribute elements evenly among cores following the strategy detailed in Section 3.3, leaving one core per node free to perform communications.

3.4.1 A Poisson problem

As a first test we solve the flat-space Poisson equation in two dimensions,

$$-\partial_i \partial_i u(\boldsymbol{x}) = f(\boldsymbol{x}), \tag{3.34}$$

for the solution

$$u_{\text{analytic}}(\boldsymbol{x}) = \sin(\pi x)\sin(\pi y) \tag{3.35}$$

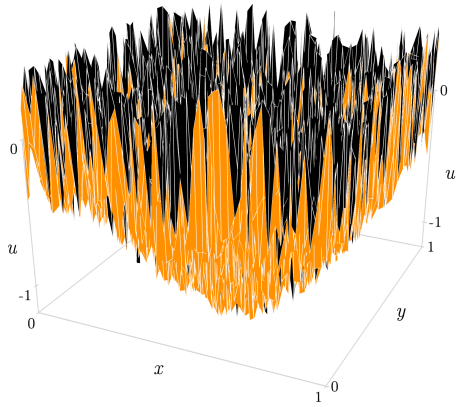
on a rectilinear domain $\Omega = [0,1]^2$ with Dirichlet boundary conditions. This problem is also studied in the context of multigrid-Schwarz methods, with slight variations, in Refs. [199, 200]. To obtain the solution (3.35) numerically we choose the fixed source $f(\boldsymbol{x}) = 2\pi^2 \sin(\pi x)\sin(\pi y)$, select homogeneous Dirichlet boundary conditions $u^{\text{b}} = 0$, and solve the DG-discretized problem (3.9) with penalty parameter $C = 1$. We have evaluated the properties of the DG-discretized operator for this problem in Ref. [1]. To assess the convergence behavior of the elliptic solver for this test problem we choose an initial guess $\underline{u}_0$, where each value is uniformly sampled from $[-0.5, 0.5]$.

[199]: Lottes and Fischer (2005), *Hybrid Multigrid/Schwarz Algorithms for the Spectral Element Method*

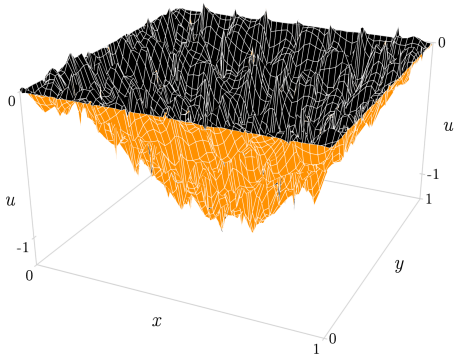
[200]: Stiller (2016), *Robust multigrid for high-order discontinuous Galerkin methods: A fast Poisson solver suitable for high-aspect ratio Cartesian grids*

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations.* Chapter 2 of this thesis.

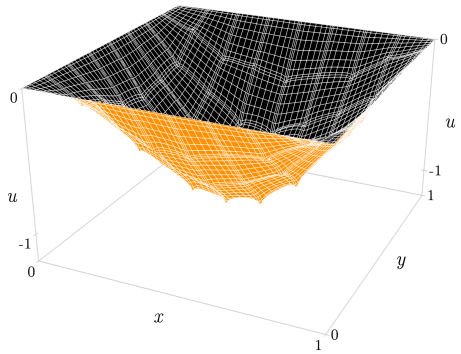
Figure 3.11 illustrates the effectiveness of our algorithm in resolving small-scale and large-scale modes in the solution. Plotted is the error to



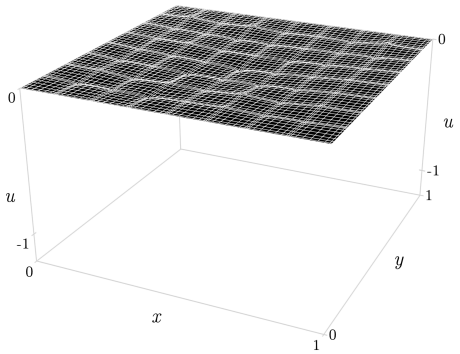
(a) Initial error $\underline{u}_0 - \underline{u}_{\text{analytic}}$



(b) Error after 6 unpreconditioned GMRES iterations



(c) Error after 6 Schwarz-smoothing iterations



(d) Error after 1 multigrid-Schwarz V-cycle

Figure 3.11: Intermediate errors of the 2D sinusoidal Poisson problem (Section 3.4.1) with different components of the elliptic solver. Panel (a) shows the error of the random initial guess. Panels (b)–(d) show the error after six applications of the linear operator.

the analytic solution, $\underline{u} - \underline{u}_{\text{analytic}}$. Figure 3.11a depicts the initial error on a computational domain that is partitioned into 8×8 quadratic elements with 9×9 grid points each. Figures 3.11b to 3.11d present the error after six applications of the linear operator, but with different components of the elliptic solver enabled. Figure 3.11b employs no preconditioning at all, thus reaches six operator applications after six iterations of the GMRES algorithm. It resolves some of the random fluctuations, but retains the large-scale sinusoidal error. Figure 3.11c preconditions every GMRES iteration with six Schwarz-smoothing steps, thus reaches six operator applications after a single GMRES iteration. The Schwarz smoother uses $N_{\text{overlap}} = 2$. It resolves most of the random fluctuations, but retains the large-scale error. Note that the six operator applications do not accurately reflect the computational expense to arrive at Fig. 3.11c, because a significant amount of work is spent on the subdomain solves. We employ the explicit-inverse subdomain solver directly here to solve subdomain problems because the Laplacian-approximation preconditioner is redundant for a pure Poisson problem (see Section 3.3.5), but note that the subdomain solver has no effect on the results depicted in Fig. 3.11 as long as it is sufficiently precise. Finally, Fig. 3.11d preconditions every GMRES iteration with a single four-level multigrid-Schwarz V-cycle. The V-cycle employs three Schwarz presmoothing and postsmoothing steps on every level, thus reaches six operator applications on the finest grid after a

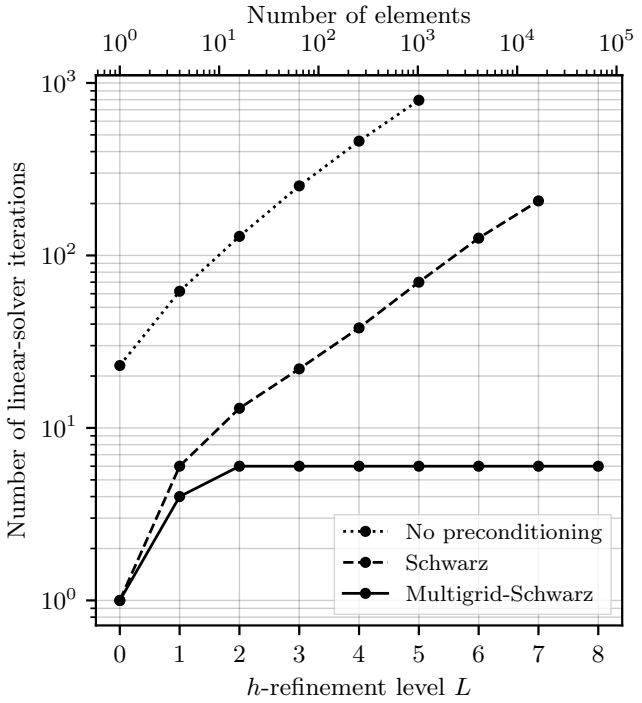


Figure 3.12: Number of linear-solver iterations for the Poisson problem (Section 3.4.1). The multigrid-Schwarz preconditioner achieves scale independence.

single GMRES iteration. Again, the number of operator applications is not entirely representative of the computational expense because it disregards the work done on coarser levels. The V-cycle successfully resolves the large-scale error.

Figure 3.12 presents the number of GMRES iterations that the elliptic solver needs to reduce the magnitude of the residual by a factor of 10^{10}, for a series of h-refined domains. We construct h-refinement levels L by repeatedly splitting all elements in the rectangular domain in two along both dimensions. All elements have 6×6 grid points. Shown in Fig. 3.12 are an unpreconditioned GMRES algorithm, a GMRES algorithm preconditioned with three Schwarz-smoothing steps per iteration, and a GMRES algorithm preconditioned with one multigrid-Schwarz V-cycle per iteration. The number of multigrid levels is equal to the number L of refinement levels, so that the coarsest level always covers the entire domain with a single element. Every level runs three presmoothing and postsmoothing steps, and subdomains have $N_{\text{overlap}} = 2$. The Schwarz preconditioning alone reduces the number of iterations by a factor of ~ 10, but does not affect the scaling with element size. However, the multigrid-Schwarz preconditioning removes the scaling entirely, meaning the number of GMRES iterations remains constant even when the domain is partitioned into more and smaller elements. The multigrid algorithm achieves this scale independence because it supports the iterative solve with information from coarser grids, including large-scale modes in the solution that span the entire domain. Each preconditioned iteration is typically more computationally expensive than an unpreconditioned iteration, but the preconditioner reduces the number of iterations such that the solve completes faster overall.[11] We find that even for the simple Poisson problem the unpreconditioned algorithm becomes prohibitively slow around $\mathcal{O}(10^3)$ elements ($L = 5$), approaching an hour of runtime and the memory capacity of our ten compute nodes. In contrast, the

11: Note that the cost of unpreconditioned GMRES iterations is eventually dominated by the orthogonalization procedure (see Section 3.3.2), which slows down the unpreconditioned solve significantly at large iteration counts. This effect can be remedied by *restarting* GMRES variants, but at the cost of possible stagnation. See Sec. 6.5.5 in Saad [127] for a discussion. Conjugate gradient algorithms avoid this issue for symmetric linear operators.

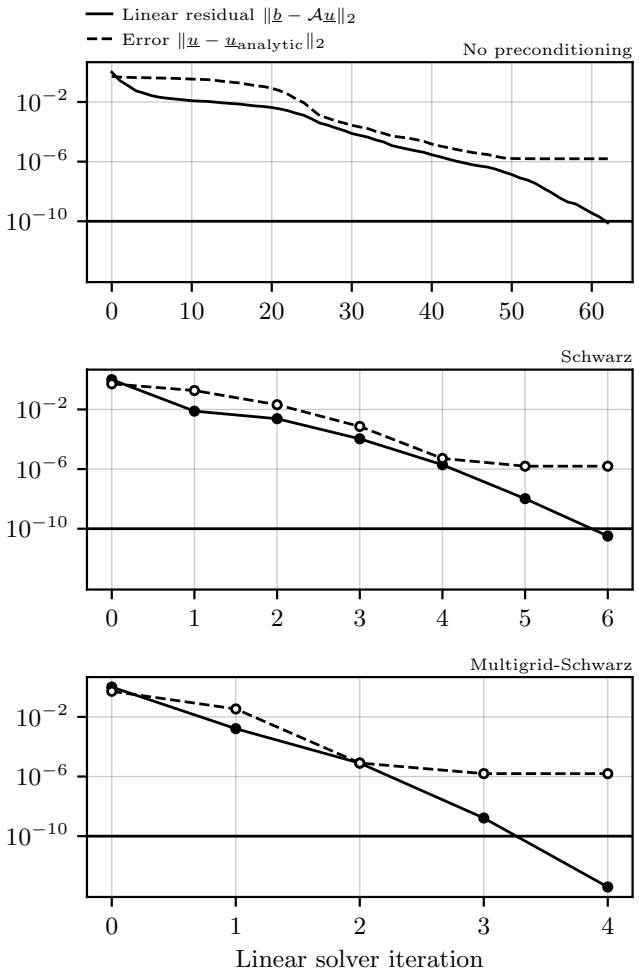


Figure 3.13: Convergence of the elliptic solver for the linear Poisson problem with h-refinement level $L = 1$. The solid line shows the relative linear-solver residual magnitude $\|\underline{b} - \mathcal{A}\underline{u}\|_2$, and the dashed line shows the error to the analytic solution, $\|\underline{u} - \underline{u}_{\text{analytic}}\|_2$ as a root mean square over grid points, which approaches the DG discretization error.

Schwarz preconditioner reduces the runtime to solve the same problem to below one minute, and the multigrid-Schwarz preconditioner reduces the runtime to only three seconds. Crucial to achieving these runtimes at high resolution are the parallelization properties of the algorithms. In particular, the additional computational expense that the preconditioner spends Schwarz-smoothing all multigrid levels is parallelizable within each level. The following test problem 3.4.2 explores the parallelization in greater detail.

Figure 3.13 gives a detailed insight into the convergence behavior of the elliptic solver for the $L = 1$ configuration. Presented is both the linear-solver residual magnitude $\|\underline{b} - \mathcal{A}\underline{u}\|_2$, and the error to the analytic solution, $\|\underline{u} - \underline{u}_{\text{analytic}}\|_2$. The linear-solver residual (solid line) is being reduced by a factor of 10^{10} by the GMRES algorithm, equipped with the three different preconditioning configurations explored in Fig. 3.12. With no preconditioning, the convergence stagnates until large-scale modes in the solution are resolved (see also Fig. 3.11). The Schwarz preconditioner reduces the number of iterations by about an order of magnitude, and the multigrid-Schwarz preconditioner achieves clean exponential convergence. The error to the analytic solution (dashed line) follows the convergence of the residual magnitude. Once the discrete problem $\mathcal{A}\underline{u} = \underline{b}$, Eq. (3.1), is solved to sufficient precision, the remaining error $\underline{u} - \underline{u}_{\text{analytic}}$ is the DG discretization error. It is independent of

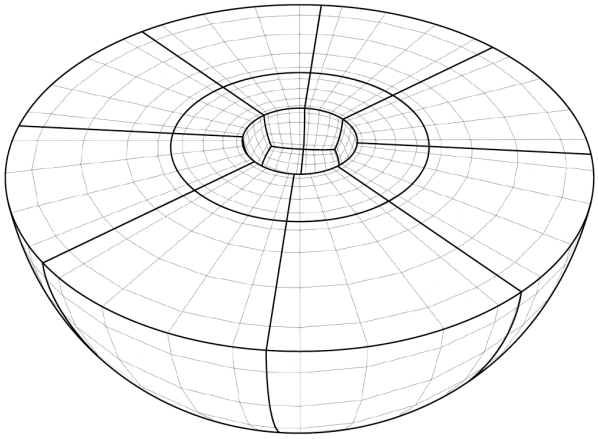


Figure 3.14: A cut through the uniformly-refined spherical-shell domain used in the black hole problem (Section 3.4.2). The domain consists of six wedges with a logarithmic radial coordinate map enveloping an excised sphere. In this example each wedge is isotropically h-refined once, i.e., split once in all three dimensions, resulting in a total of 48 elements. Note the elements are split in half along their logical axes, so the element size scales logarithmically in radial direction just like the distribution of grid points within the elements. Each element has six grid point per dimension, so fields are represented as polynomials of degree five.

the computational technique used to solve the discrete problem, and determined entirely by the discretization scheme on the computational grid, as summarized in Section 3.2 and detailed in Ref. [1].[12]

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

12: For a study of the DG discretization error for this problem see Fig. 2.7, where the configuration solved in Fig. 3.13 is circled.

3.4.2 A black hole in general relativity

Next, we solve a general-relativistic problem involving a black hole. Specifically, we solve the Einstein constraint equations in the XCTS formulation, Eq. (3.4), for a Schwarzschild black hole in Kerr-Schild coordinates. To this end we set the conformal metric and the trace of the extrinsic curvature to their respective Kerr-Schild quantities,

$$\bar{\gamma}_{ij} = \delta_{ij} + \frac{2M}{r} l_i l_j \tag{3.36a}$$

and

$$K = \frac{2M\alpha^3}{r^2}\left(1 + \frac{3M}{r}\right), \tag{3.36b}$$

where M is the mass parameter, $r = \sqrt{x^2 + y^2 + z^2}$ is the Euclidean coordinate distance, and $l^i = l_i = x^i/r$.[13] The time-derivative quantities $\bar{u}_{ij}$ and $\partial_t K$ in the XCTS equations (3.4) vanish, as do the matter sources ρ, S, and S^i. With these background quantities specified, the solution to the XCTS equations is

13: See Table 2.1 in Baumgarte and Shapiro [17].

[17]: Baumgarte and Shapiro (2010), *Numerical Relativity: Solving Einsteins Equations on the Computer*

$$\psi = 1, \tag{3.37a}$$

$$\alpha = \left(1 + \frac{2M}{r}\right)^{-1/2}, \tag{3.37b}$$

$$\beta^i = \frac{2M}{r}\alpha^2 l^i. \tag{3.37c}$$

Note that we have chosen a conformal decomposition with $\psi = 1$ here, but other choices of ψ and $\bar{\gamma}_{ij}$ that keep the spatial metric $\gamma_{ij} = \psi^4\bar{\gamma}_{ij}$ invariant are equally admissable.

We solve the XCTS equations numerically for the conformal factor ψ, the product $\alpha\psi$, and the shift β^i. The conformal metric $\bar{\gamma}_{ij}$ and the trace of the extrinsic curvature, K, are background quantities that remain fixed

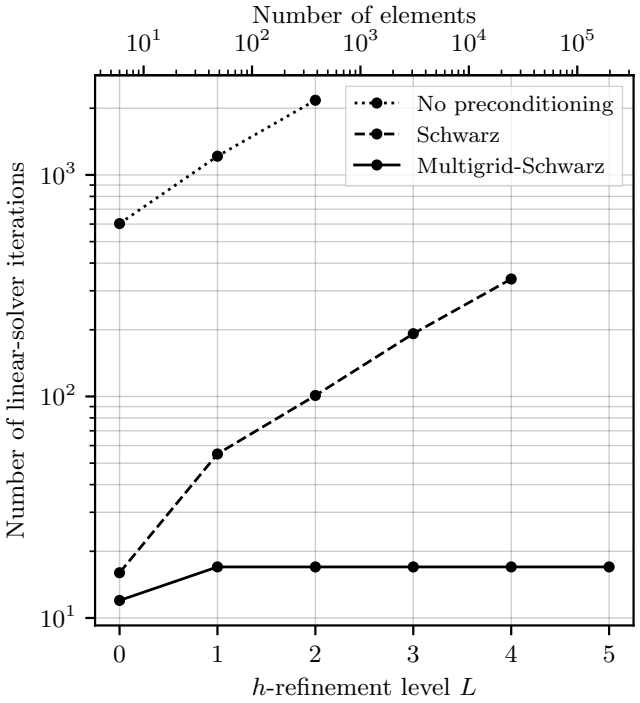


Figure 3.15: Number of linear-solver iterations for the black hole problem (Section 3.4.2). The multigrid-Schwarz preconditioner achieves scale independence. The $L = 1$ configuration (48 elements) is pictured in Fig. 3.14.

throughout the solve. Note that for this test problem the conformal metric $\bar{\gamma}_{ij}$ is not flat, resulting in a problem formulated on a curved manifold.

We employ the DG scheme (3.9) with penalty parameter $C = 1$ to discretize the XCTS equations (3.4) on a three-dimensional spherical-shell domain, as illustrated in Fig. 3.14. The domain envelops an excised sphere that represents the black hole, so it has an inner and an outer external boundary that require boundary conditions. To obtain the Schwarzschild solution in Kerr-Schild coordinates we impose Eqs. (3.37a) to (3.37c) as Dirichlet conditions at the outer boundary of the spherical shell at $r = 10M$. We place the inner radius of the spherical shell at $r = 2M$ and impose mixed Dirichlet and Neumann conditions at the inner boundary. Specifically, we impose the Neumann condition $n^i \partial_i \psi = 0$ on the conformal factor, and Eqs. (3.37b) to (3.37c) as Dirichlet conditions on the lapse and shift. The reason for this choice is to mimic apparent-horizon boundary conditions, as employed in the following test problem (Section 3.4.3). Choosing apparent-horizon boundary conditions for the Kerr-Schild problem is also possible, but requires either an initial guess close to the solution to converge, or a conformal decomposition different from $\psi = 1$. The reason is the strong nonlinearity in the apparent-horizon boundary conditions that takes the solution away from $\psi_0 = 1$ initially. We have confirmed this behavior of the XCTS equations with the `SpEC` code, and have presented the convergence of the DG discretization error with apparent-horizon boundary conditions for the Kerr-Schild problem in Ref. [1]. With the simpler Dirichlet and Neumann boundary condition we can seed the elliptic solver with a flat initial guess, i.e., $\psi_0 = 1$, $\alpha_0 = 1$ and $\beta_0^i = 0$, which allows for better control of the test problem.[14]

To assess the convergence behavior of the elliptic solver for this problem we successively h refine the wedges of the spherical-shell domain into more and smaller elements, each with six grid points per dimension.

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

14: I discuss this behavior of the XCTS equations in detail in Section 4.1.1.

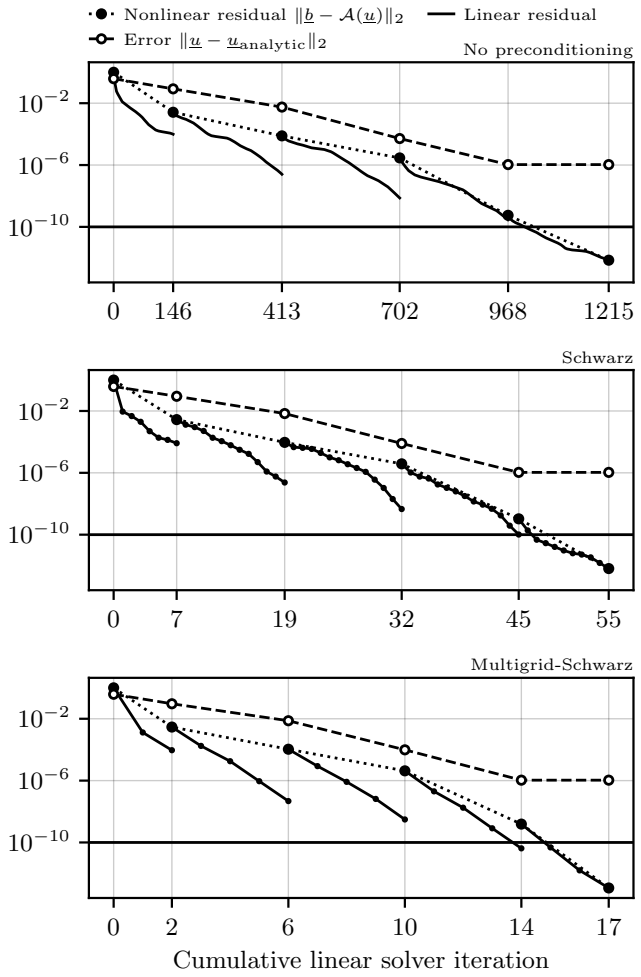


Figure 3.16: Convergence of the Newton-Krylov elliptic solver for the black hole problem with h-refinement level $L = 1$. The dotted line shows the relative residual magnitude of the nonlinear solver, $\|\underline{b} - \mathcal{A}\underline{u}\|_2$, which is driven by a linear solve in every iteration (solid lines, see Eq. (3.16)). The dashed line shows the error to the analytic solution, $\|\underline{u} - \underline{u}_{\text{analytic}}\|_2$ as a root mean square over all five variables of the XCTS equations, $\{\psi, \alpha\psi, \beta^i\}$, and over grid points. It approaches the DG discretization error.

We iterate the Newton-Raphson algorithm until the magnitude of the nonlinear residual has decreased by a factor of 10^{10}. In all configurations we have tested, the nonlinear solver needs five steps and no line-search globalization to reach the target residual. The linear solver is configured to solve the linearized problem, Eq. (3.16), by reducing its residual magnitude by a factor of 10^4. Schwarz subdomains have $N_{\text{overlap}} = 2$, and we run three Schwarz presmoothing and postsmoothing iterations on every multigrid level, including the coarsest. Figure 3.15 presents the total number of linear-solver iterations accumulated over the five nonlinear solver steps. Just like we found for the simple Poisson problem in Fig. 3.12, the multigrid-Schwarz preconditioner achieves scale-independent iteration counts under h refinement.

Figure 3.16 presents the convergence behavior of the elliptic solver for the $L = 1$ configuration (pictured in Fig. 3.14) in detail. The convergence of the nonlinear residual magnitude (dotted line) is independent of the preconditioner chosen for the linear solver in each iteration (solid lines), since the linearized problems are solved to sufficient accuracy (10^{-4}). Similar to the Poisson problem in Fig. 3.13, the multigrid-Schwarz preconditioning achieves clean exponential convergence, reducing the linear residual by an order of magnitude per iteration. The nonlinear residual magnitude decreases slowly at first, when the fields $\underline{u}$ are still far from the solution, and begins to converge quadratically, following the

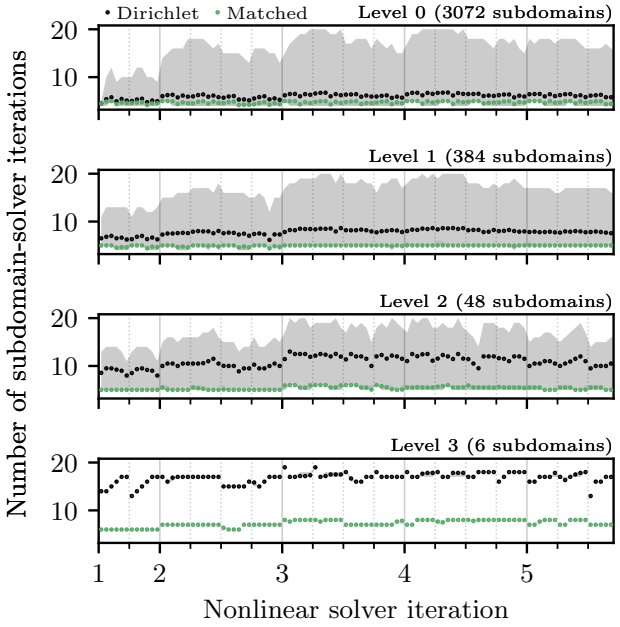


Figure 3.17: Number of subdomain-solver iterations for the black hole problem (Section 3.4.2) with the Laplacian-approximation preconditioner. The domain is isotropically h-refined thrice, so the solve involves four multigrid levels. Dots illustrate the average across all subdomains on the level, and shaded regions the smallest and largest number of iterations. When approximating all equations with a Dirichlet-Laplacian (black), subdomains facing the inner excision boundary (see Fig. 3.14) require more iterations than the average. Matching the Laplacian boundary conditions to the problem (green) reduces the iteration count and resolves the load imbalance.

linear-solver residual, once the fields are closer to the solution and hence the linearization is more accurate (see Section 3.3.1). The error to the analytic solution (dashed line) approaches the DG discretization error, as detailed in Section 3.4.1.[15]

15: For a study of the DG discretization error for this problem see Fig. 2.11, where the configuration solved in Fig. 3.16 is circled.

To solve subdomain problems here we equip the GMRES subdomain solver with the Laplacian-approximation preconditioner, and solve the five Poisson subproblems on every subdomain with the incomplete LU explicit-inverse solver (see Section 3.3.5). Figure 3.17 illustrates the importance of matching the boundary conditions of the approximate Laplacian to the original problem. When we approximate all five tensor components of the original XCTS problem with a Dirichlet-Laplacian, ignoring that we impose Neumann-type boundary condition on ψ at the inner boundary, some subdomains require a significantly larger number of subdomain-solver iterations than others. We have confirmed that these subdomains face the inner boundary of the spherical shell. When we use a Laplacian approximation with matching boundary-condition type for these subdomains, they need no more subdomain-solver iterations than interior subdomains. Specifically, the subdomain preconditioner constructs a Poisson operator matrix with homogeneous Neumann boundary conditions to apply to the conformal-factor component of the equations, and another with homogeneous Dirichlet boundary conditions to apply to the remaining four tensor components. Therefore, subdomains that face the inner boundary of the spherical shell domain build and cache two inverse matrices, and all other subdomains build and cache a single inverse matrix, in the form of an incomplete LU decomposition. Furthermore, when the Laplacian-approximation preconditioner takes the type of boundary conditions into account, we find that it is sufficiently precise so we can limit the number of subdomain-solver iterations to a fixed number. This further balances the load between elements, decreasing runtime significantly in our tests. Therefore, in the following we always limit the number of subdomain-solver iterations to three. With this strategy we find a reduction in runtime of about 50 % compared to

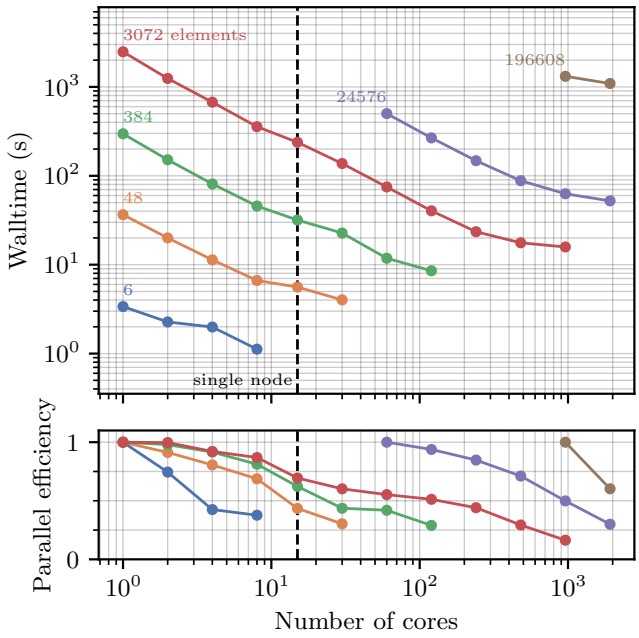


Figure 3.18: Parallel scaling of the black hole problem (Section 3.4.2).

the naive Dirichlet-Laplacian approximation.

Figure 3.18 presents the wall time and parallel efficiency of the elliptic solves for the black hole problem on up to 2048 cores, which approaches the capacity of our local computing cluster. We split the domain into more and smaller elements, keeping the number of grid points in each element constant at six per dimension, and solve each configuration on a variable number of cores. These configurations are increasingly expensive to solve, involving up to 42 million grid points, or ~200 million degrees of freedom. They all complete in at most a few minutes of wall time by scaling up to a few thousand cores, until they reach the capacity of our cluster. We compute their parallel efficiency as

$$\text{Parallel efficiency} = \frac{t_\text{serial}}{t_\text{CPU}}, \tag{3.38}$$

where $t_\text{CPU} = N_\text{cores}\, t_\text{wall}$ is the CPU time of a run, i.e., the product of wall time and the number of cores, and t_serial is the wall time of the configuration runing on a single core. Since configurations with 24576 elements and more did not complete on a single core in the allotted time of two hours, we approximate t_serial with the CPU time of the run with the lowest number of cores for these configurations, meaning that they begin at a fiducial parallel efficiency of one. The parallel efficiency decreases when the number of elements per core becomes small and falls below 25 % once each core holds only a few elements.

Figure 3.18 also shows that the parallel efficiency decreases more steeply when filling up a single node, than it does when we begin to allocate multiple nodes. We take this behavior as an indication that shared hardware resources on a node currently limit our parallel efficiency, which is an issue also found in Ref. [201]. We have confirmed this hypothesis by running a selection of configurations on the same number of cores, but distributed over more nodes, so each node is only partially subscribed, and found that runs speed up significantly. We intend to address this

[201]: AlOnazi, Markomanolis, and Keyes (2017), *Asynchronous Task-Based Parallelization of Algebraic Multigrid*

issue in future optimizations of the elliptic solver. Possible resolutions include better use of CPU caches, e.g., through a contiguous layout of data on subdomains, or a shared-memory OpenMP parallelization of subdomain solves, so the cores of a node are working on a smaller amount of data at any given time. The parallel efficiency also decreases once we reach the capacity of our cluster, at which point we expect that communications spanning the full cluster dominate the computational expense. We intend to test the parallel scaling on larger clusters with more cores per node in the future. We also plan to investigate the effect of hyperthreading on the parallel efficiency of the elliptic solver.

3.4.3 A black hole binary

Finally, we solve a classic black hole binary (BBH) initial data problem, which stands at the beginning of every BBH simulation performed with the `SpEC` code. Again, we solve the full Einstein constraint system in the XCTS formulation, Eq. (3.4), but now we choose background quantities and boundary conditions that represent two black holes in orbit. Following the formalism for *superposed Kerr-Schild* initial data, e.g., laid out in Ref. [75, 78], we set the conformal metric and the trace of the extrinsic curvature to the superpositions

[75]: Lovelace et al. (2008), *Binary-black-hole initial data with nearly-extremal spins*
[78]: Varma, Scheel, and Pfeiffer (2018), *Comparison of binary black hole initial data sets*

$$\bar{\gamma}_{ij} = \delta_{ij} + \sum_{n=1}^{2} e^{-r_n^2/w_n^2} \left(\gamma_{ij}^{(n)} - \delta_{ij}\right) \tag{3.39a}$$

and

$$K = \sum_{n=1}^{2} e^{-r_n^2/w_n^2} K^{(n)}, \tag{3.39b}$$

where $\gamma_{ij}^{(n)}$ and $K^{(n)}$ are the conformal metric and extrinsic-curvature trace of two isolated Schwarzschild black holes in Kerr-Schild coordinates as given in Eq. (3.36). They have mass parameters M_n and are centered at coordinates $\boldsymbol{C}_n$, with r_n being the Euclidean coordinate distance from either center. The superpositions are modulated by two Gaussians with widths w_n. The time-derivative quantities $\bar{u}_{ij}$ and $\partial_t K$ in the XCTS equations (3.4) vanish, as do the matter sources ρ, S and S^i.

To handle orbital motion we split the shift in a *background* and an *excess* contribution [139],

[139]: Pfeiffer et al. (2003), *A multidomain spectral method for solving elliptic equations*

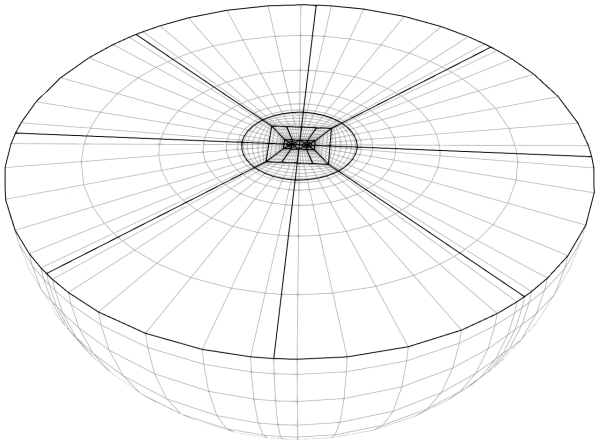
$$\beta^i = \beta^i_{\text{background}} + \beta^i_{\text{excess}}, \tag{3.40}$$

and choose the background shift

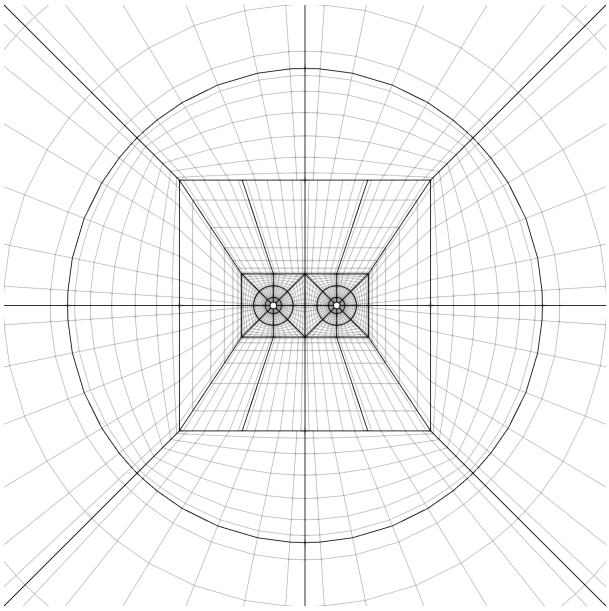
$$\beta^i_{\text{background}} = (\boldsymbol{\Omega}_0 \times \boldsymbol{x})^i, \tag{3.41}$$

where $\boldsymbol{\Omega}_0$ is the orbital angular velocity. We insert Eq. (3.40) in the XCTS equations (3.4) and henceforth solve them for β^i_{excess}, instead of β^i.

We solve the XCTS equations on the domain depicted in Fig. 3.19. It has two excised spheres with radius $2M_n$ that are centered at $\boldsymbol{C}_n$, and correspond to the two black holes, and an outer spherical boundary at finite radius R. We impose boundary conditions on these three boundaries as follows. At the outer spherical boundary of the domain we impose



(a) Black hole binary domain



(b) Close-up

Figure 3.19: A cut through the three-dimensional black hole binary domain used in Section 3.4.3. It involves two excised spheres centered at $\boldsymbol{C}_n$ along the x-axis and extends to a spherical outer surface at radius R. The domain is h-refined such that spherical wedges have equal angular size, so the cube-to-sphere boundary is nonconforming. All elements in this picture have eight angular grid points, and $\{7, 8, 8, 9, 11, 11\}$ radial grid points in the layers ordered from outermost to innermost.

asymptotic flatness,

$$\psi = 1, \quad \alpha\psi = 1, \quad \beta^i_{\text{excess}} = 0. \tag{3.42}$$

Since the outer boundary is at a finite radius, the solution will only be approximately asymptotically flat. On the two excision boundaries we impose (nonspinning) quasiequilibrium apparent-horizon boundary conditions [113]

[113]: Cook and Pfeiffer (2004), *Excision boundary conditions for black hole initial data*

$$\begin{aligned} \bar{s}^k \partial_k \psi = &-\frac{\psi^3}{8\alpha} \bar{s}_i \bar{s}_j \left((\bar{L}\beta)^{ij} - \bar{u}^{ij} \right) \\ &- \frac{\psi}{4} \bar{m}^{ij} \bar{\nabla}_i \bar{s}_j + \frac{1}{6} K \psi^3, \end{aligned} \tag{3.43a}$$

$$\beta^i = \frac{\alpha}{\psi^2} \bar{s}^i, \tag{3.43b}$$

where $\bar{m}^{ij} = \bar{\gamma}^{ij} - \bar{s}^i \bar{s}^j$ (see Section 1.2.3). Here, $\bar{s}_i = -n_i = \psi^{-2} s_i$ is the conformal surface normal to the apparent horizon, which is opposite the normal to the domain boundary since both are normalized with the conformal metric. For the lapse we choose to impose the isolated solution,

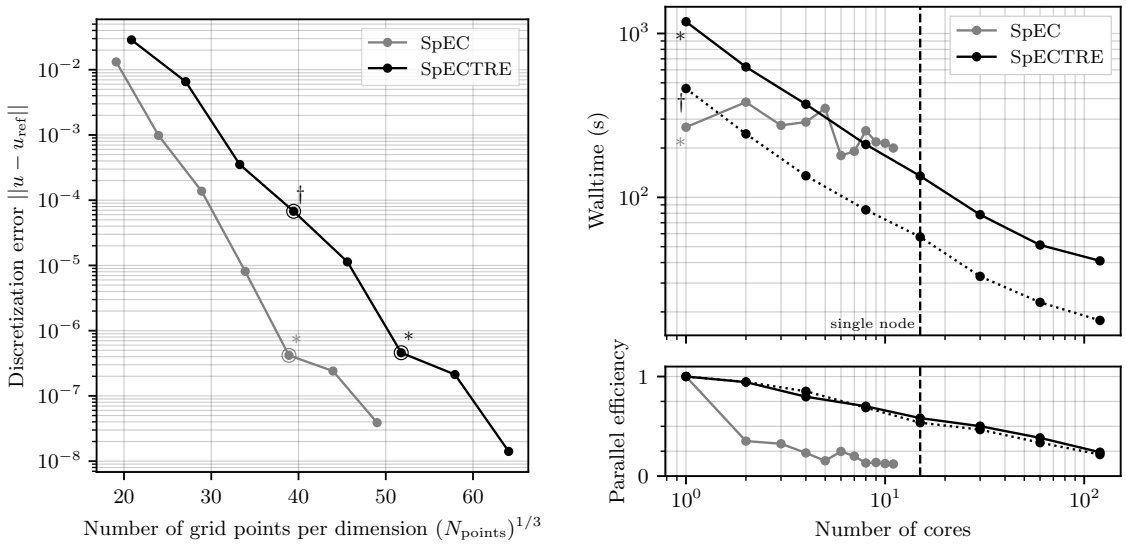


Figure 3.20: Comparison of the BBH initial data problem (Section 3.4.3) solved with our new elliptic solver in `SpECTRE` (black), and with the `SpEC` elliptic solver (gray). *Left:* Both codes converge exponentially with resolution. `SpECTRE` needs about 30 % more grid points per dimension than `SpEC` to reach the same accuracy. *Right:* Parallel scaling of both codes. The `SpEC` elliptic solver scales to at most eleven cores and reaches a speedup of at most a factor of two compared to the single-core runtime. Our new elliptic solver in `SpECTRE` is faster than `SpEC` on eight cores, and scales the problem reliably to 120 cores, at which point it is seven times faster than `SpEC`'s single-core runtime. The dotted line corresponds to a configuration with the same number of grid points as `SpEC` (but lower accuracy), which is faster than `SpEC` on only two cores.

Eq. (3.37b), as Dirichlet conditions at both excision surfaces. Note that this choice differs slightly from Ref. [78], where the *superposed* isolated solutions are imposed on the lapse at both excision surfaces.

[78]: Varma, Scheel, and Pfeiffer (2018), *Comparison of binary black hole initial data sets*

To assess the accuracy and parallel performance of the elliptic solver for the BBH initial data problem we solve the same scenario with the `SpEC` [55, 139] code. In `SpEC` we successively increment the resolution from `Lev0` to `Lev6`, which correspond to domain configurations determined with an adaptive mesh-refinement (AMR) algorithm. In `SpECTRE` we simply increment the number of grid points in all dimensions of all elements by one from each resolution to the next, based on the domain depicted in Fig. 3.19. To compare the solution between the two codes, we interpolate all five fields $u_A = \{\psi, \alpha\psi, \beta^i_{\text{excess}}\}$ to a set of sample points $\boldsymbol{x}_m$. We do the same for a very high-resolution run with `SpECTRE` that we use as reference, $u_{A,\text{ref}}$, for which we have split every element in the domain in two along all three dimensions. Then, we compute the discretization error for all `SpEC` and `SpECTRE` solutions as an L_2-norm of the difference to the high-resolution reference run over all fields and sample points,

[55]: Spectral Einstein Code (SpEC), `black-holes.org/code/SpEC`
[139]: Pfeiffer et al. (2003), *A multidomain spectral method for solving elliptic equations*

$$\|u - u_{\text{ref}}\| \coloneqq \left(\sum_{A,m} \left(u_A(\boldsymbol{x}_m) - u_{A,\text{ref}}(\boldsymbol{x}_m) \right)^2 \right)^{1/2}. \tag{3.44}$$

We have chosen $M_n = 0.4229$, $\boldsymbol{C}_n = (\pm 8, 0, 0)$, $\Omega_0 = 0.0144$, $w_n = 4.8$, $R = 300$, and sample points along the x-axis at $x_1 = 8.846$ (near horizon), $x_2 = 0$ (origin) and $x_3 = 100$ (far field) here. This configuration coincides with our convergence study in Ref. [1], where we list the reference values $u_{A,\text{ref}}(\boldsymbol{x}_m)$ at the interpolation points explicitly (Table 2.2).

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

Figure 3.20 compares the performance of the BBH initial data problem

with the SpEC code. Both SpEC and SpECTRE converge exponentially with resolution, since SpEC employs a spectral scheme and SpECTRE a DG scheme under p refinement. SpECTRE currently needs about 30 % more grid points per dimension to achieve the same accuracy as SpEC. To an extent this is to be expected, since we split the domain into more elements than SpEC does and hence have more shared element boundaries with duplicate points. In particular, SpEC employs shells with spherical basis functions that avoid duplicate points in angular directions altogether. While the SpEC elliptic solver always decomposes the domain into eleven subdomains, each with up to 34 grid points per dimension in our test, our domain in SpECTRE has 232 elements with up to 13 grid points per dimension. However, neither is our BBH domain in SpECTRE optimized as well as SpEC's yet, nor have we refined it with an AMR algorithm. We are planning to do both in future work. Furthermore, SpEC's initial data domain involves overlapping patches to enable matching conditions for its spectral scheme, which our DG scheme in SpECTRE does not need. Therefore, we expect to achieve domain configurations that come closer to SpEC in their number of grid points with future optimizations.

The right panel in Fig. 3.20 demonstrates the superior parallel performance that our new elliptic solver achieves over SpEC's. We choose the runs marked with $*$ in the left panel because they solve the BBH problem to comparable accuracy. We scale these runs to an increasing number of cores and measure the wall time for the elliptic solves to complete. Since the SpEC initial data domain is composed of eleven subdomains, it can parallelize to at most eleven cores. The runtime decreases by a factor of about 1.5 to 2 when the solve is distributed to multiple cores, but shows little reliable scaling. Some SpEC configurations at higher resolutions have shown slightly better parallel performance, but none that exceeded a factor of about two in speedup compared to the single-core runtime. Our new elliptic solver in SpECTRE, on the other hand, scales reliably to 120 cores, at which point each core holds no more than two elements. On a single core it needs 1176 s where SpEC, with fewer grid points, needs only 268 s, but it overtakes SpEC on eight cores and completes in only 37 s on 120 cores. For reference we have also included a scaling test with SpECTRE that uses the same number of grid points as SpEC but does not yet reach the same accuracy (marked with †). It overtakes SpEC on two cores and completes in only 14 s on 120 cores. The configuration represents a potential improvement in the domain decomposition with future optimizations. We find the parallel efficiency for the BBH configurations behaves similarly to the single black hole configurations we investigated in Section 3.4.2.

3.5 Conclusion and future work

We have presented a new solver for elliptic partial differential equations that is designed to parallelize on computing clusters. It is based on discontinuous Galerkin (DG) methods and task-based parallel iterative algorithms. We have shown that our solver is capable of parallelizing elliptic problems with ~200 million degrees of freedom to at least a few thousand cores. It solves a classic black hole binary (BBH) initial data problem faster than the veteran code SpEC [55] on only eight cores, and

[55]: Spectral Einstein Code (SpEC), black-holes.org/code/SpEC

in a fraction of the time when distributed to more cores on a computing cluster. The elliptic solver is implemented in the open-source `SpECTRE` [10] numerical relativity code, and the results in this article are reproducible with the supplemental input-file configurations [205].

[10]: SpECTRE, spectre-code.org

[205]: Supplemental material: input-file configurations to reproduce the results presented in this article with the SpECTRE code, `arXiv:2111.06767/anc`

So far we can solve Poisson, elasticity, puncture and XCTS problems, including BBH initial data in the superposed Kerr-Schild formalism with unequal masses, spins and negative-expansion boundary conditions [78] (in this article we only explored an equal-mass and nonspinning BBH). In the short term we are planning to add the capability to solve for binary neutron star (BNS) and black hole–neutron star (BHNS) initial data, which involve the XCTS equations coupled to the equations of hydrostatic equilibrium.

[78]: Varma, Scheel, and Pfeiffer (2018), *Comparison of binary black hole initial data sets*

A notable strength of our new elliptic solver is the multigrid-Schwarz preconditioner, which achieves iteration counts independent of the number of elements in the computational domain. Therefore, we expect our solver to scale to problems that benefit from h refinement, e.g., to resolve different length scales or to adapt the domain to features in the solution. Such problems include initial data involving neutron stars with steep gradients near the surface, equations of state with phase transitions, or simulating thermal noise in thin mirror coatings for gravitational-wave detectors [3, 167].

[3]: **Vu** et al. (2021), *High-accuracy numerical models of Brownian thermal noise in thin mirror coatings*. Chapter 5 of this thesis.
[167]: Lovelace, Demos, and Khan (2018), *Numerically modeling Brownian thermal noise in amorphous and crystalline thin coatings*

Our solver splits the computational domain into more elements than the spectral code `SpEC` to achieve superior parallelization properties. However, the larger number of elements with shared boundaries also means that we need more grid points than `SpEC` to reach the same accuracy for a BBH initial data problem. Variations of the DG scheme, such as a hybridizable DG method, can provide a possible resolution to this effect [206–208]. Even without changing the DG scheme, we expect that optimizations of our binary compact object domain can significantly reduce the number of grid points required to reach a certain accuracy. Possible domain optimizations include combining the enveloping cube and the cube-to-sphere transition into a single layer of blocks, equalizing the angular size of the enveloping wedges in a manner similar to Ref. [108], or more drastic changes that involve cylindrical or bipolar coordinate maps, such as the domain presented in Ref. [209].[16] To retain the effectiveness of the multigrid solver it is important to keep the number of blocks in the domain to a minimum under these optimizations. We have shown that our new elliptic solver reaches comparative single-core performance to `SpEC` when using the same number of grid points, with the added benefit of parallel scaling. Since every contemporary computer has multiple cores, we prioritize parallelization over single-core performance.

[206]: Cockburn, Gopalakrishnan, and Lazarov (2009), *Unified Hybridization of Discontinuous Galerkin, Mixed, and Continuous Galerkin Methods for Second Order Elliptic Problems*
[207]: Giacomini, Sevilla, and Huerta (2021), *HDGlab: An Open-Source Implementation of the Hybridisable Discontinuous Galerkin Method in MATLAB*
[208]: Muralikrishnan, Bui-Thanh, and Shadid (2020), *A multilevel approach for trace system in HDG discretizations*

[108]: Rashti et al. (2021), *Elliptica: a new pseudo-spectral code for the construction of initial data*
[209]: Buchman et al. (2012), *Simulations of unequal-mass black hole binaries with spectral methods*

16: I discuss domain optimizations since the publication of this article in Section 4.2.2.

To put the grid points of the computational domain to most effective use, adaptive mesh-refinement (AMR) techniques will be essential. All components of the elliptic solver, including the DG discretization, the multigrid algorithm, and the Schwarz subdomains, already support hp-refined domains. The refinement can be anisotropic, meaning elements can be split or differ in their polynomial degree along each dimension independently. A major subject of future work will be the development of an AMR scheme that adjusts the refinement during the elliptic solve automatically based on a local error estimate, distributing resolution to regions in the domain where it is most needed.

Along with AMR, we expect load balancing to become increasingly important. We currently approximately load balance the elliptic solver at the beginning of the program based on the number of grid points in each element. `Charm++`, and hence `SpECTRE`, also support dynamic load-balancing operations that migrate elements between cores periodically, or at specific points in the algorithm. `Charm++` provides a variety of load-balancing algorithms that may take metrics such as runtime measurements, communication cost and the network topology into account. When the computational load on elements changes due to p-AMR, or when elements get created and destroyed due to h-AMR, we intend to invoke load balancing to improve the parallel performance of the elliptic solves.

The elliptic solver algorithms can be improved in many ways. The multigrid solver may benefit from an additive variant of the algorithm, which smooths every level independently and combines the solutions [201]. An additive multigrid algorithm has better parallelization properties than the multiplicative algorithm that we employ in this article, since coarse grids do not need to wait for fine grids to send data before the coarse-grid smoothing can proceed. However, the additive multigrid algorithm typically requires more iterations to converge than the multiplicative. Furthermore, multigrid patterns other than the standard V-cycle may accelerate convergence, such as a W-cycle or F-cycle pattern [198].

[201]: AlOnazi, Markomanolis, and Keyes (2017), *Asynchronous Task-Based Parallelization of Algebraic Multigrid*

[198]: Briggs, Henson, and McCormick (2000), *A Multigrid Tutorial*

Schwarz solvers also come in many variations, e.g., involving face-centered subdomains, that we have not explored in this article. Our element-centered subdomains that eliminate corner and edge neighbors have served well for our DG-discretized problems so far, and we have focused on accelerating the subdomain solves with suitable preconditioners. Faster explicit-construction and approximate-inversion techniques for the subproblems in the Laplacian-approximation preconditioner have the greatest potential to speed up the elliptic solver. Possibilities include constructing matrix representations analytically, either from the DG scheme or from an approximate finite-difference scheme, and faster methods to solve the subproblems than the incomplete LU technique we currently employ.

A possible avenue for a more drastic improvement of the elliptic solver algorithm is to replace the multigrid-Schwarz preconditioner, or parts of it, altogether. For example, recent developments in the field of physics-informed neural networks (PINNs) suggest that hybrid strategies, combining a traditional linear solver with a PINN, can be very effective [210, 211]. Hence, an intriguing prospect for accelerating elliptic solves in numerical relativity is to combine our Newton-Krylov algorithm with a PINN preconditioner, use the PINN as a smoother on multigrids, or use it to precondition Schwarz subdomain solves.

[210]: Markidis (2021), *The Old and the New: Can Physics-Informed Deep-Learning Replace Traditional Linear Solvers?*
[211]: Guidetti et al. (2021), *dNNsolve: an efficient NN-based PDE solver*

Looking ahead, fast, scalable and highly-parallel elliptic solves in numerical relativity not only have the potential to accelerate initial data construction to seed high-resolution simulations of general-relativistic scenarios, and at extreme physical parameters, but they may also support evolutions. For example, some gauge constraints can be formulated as elliptic equations, and solving them alongside an evolution can allow the choice of beneficial coordinates, such as maximal slicing [17]. The apparent-horizon condition is also an elliptic equation, though current

[17]: Baumgarte and Shapiro (2010), *Numerical Relativity: Solving Einsteins Equations on the Computer*

NR codes typically find apparent horizons with a parabolic relaxation method [115]. Some NR codes employ a constrained-evolution scheme, which evolves the system in time through a series of elliptic solves, or employ implicit-explicit (IMEX) evolution schemes [122]. Lastly, Einstein-Vlasov systems for collisionless matter involve elliptic equations, as do simulations that involve solving a Poisson equation alongside an evolution, such as simulations of self-gravitating protoplanetary disks [212–214]. Currently, elliptic solvers are rarely applied to solve any of these problems alongside an evolution because they are too costly. Fast elliptic solves have the potential to enable these applications.

[115]: Gundlach (1998), *Pseudospectral apparent horizon finders: An Efficient new algorithm*

[122]: Lau, Lovelace, and Pfeiffer (2011), *Implicit-explicit (IMEX) evolution of single black holes*

[212]: Deng, Mayer, and Latter (2020), *Global Simulations of Self-gravitating Magnetized Protoplanetary Disks*

[213]: Hopkins (2015), *A new class of accurate, mesh-free hydrodynamic simulation methods*

[214]: Enzo (2014), *Enzo: An Adaptive Mesh Refinement Code for Astrophysics*

Acknowledgments

The authors thank Tim Dietrich, Francois Foucart, and Hannes Rüter for helpful discussions. N. V. also thanks the Cornell Center for Astrophysics and Planetary Science and TAPIR at Caltech for the hospitality and financial support during research stays. Computations were performed with the `SpECTRE` code [10] on the `Minerva` cluster at the Max Planck Institute for Gravitational Physics. `Charm++` [141] was developed by the Parallel Programming Laboratory in the Department of Computer Science at the University of Illinois at Urbana-Champaign. The figures in this article were produced with `dgpy` [11], `matplotlib` [172, 173], `TikZ` [174] and `ParaView` [175]. This work was supported in part by the Sherman Fairchild Foundation and by NSF Grants No. PHY-2011961, No. PHY-2011968, and No. OAC-1931266 at Caltech, and NSF Grants No. PHY-1912081 and No. OAC-1931280 at Cornell. G. L. is pleased to acknowledge support from the NSF through Grants No. PHY-1654359 and No. AST-1559694 and from Nicholas and Lee Begovich and the Dan Black Family Trust.

[10]: SpECTRE, spectre-code.org

[141]: The Charm++ Parallel Programming System, `https://charm.cs.illinois.edu`

[11]: dgpy, 10.5281/zenodo.5086181

[172]: Hunter (2007), *Matplotlib: A 2D graphics environment*

[173]: matplotlib, 10.5281/zenodo.4268928

[174]: pgf - A Portable Graphic Format for TeX, `github:pgf-tikz/pgf`

[175]: Ahrens, Geveci, and Law (2005), *ParaView: An End-User Tool for Large-Data Visualization*

Initial data with black holes and neutron stars

4

Discussion chapter

In this chapter I discuss the applicability of the discontinuous Galerkin discretization scheme (Chapter 2 and Ref. [1]) and my new task-based parallel elliptic solver (Chapter 3 and Ref. [2]) to the initial data problem in numerical relativity. This is not the only target application of the elliptic solver (I present another in Chapter 5, and have discussed further possible applications in Chapter 1), but it is certainly the primary application, and the one that has primarily driven the development of the SpECTRE code thus far. The results in this chapter are as yet unpublished. They extend the test cases in Chapters 2 and 3 to ready-to-evolve initial data for simulations of binary black holes and neutron stars, with the goal to seed our general-relativistic evolutions with the SpECTRE code.[a]

[a]At the time of writing, the SpECTRE code is capable of evolving BBH systems in the generalized harmonic (GH) formulation using pregenerated control-system data from a preceding SpEC simulation, which is used to control the position and shape of the excision surfaces throughout the SpECTRE evolution. Therefore, these BBH evolutions currently rely on SpEC initial data interpolated to the SpECTRE grid. Work on control systems in SpECTRE is underway and will enable evolutions of the BBH initial data sets presented in this chapter, making BBH evolutions in SpECTRE independent of SpEC. Work on GRMHD evolutions in SpECTRE is also underway and will enable the evolution of initial data involving neutron stars. See Ref. [7] for recent progress on GRMHD evolutions in SpECTRE.

In this chapter, I solve the elliptic Einstein constraint equations in the XCTS formalism (see Section 1.2.3) to generate slices of general-relativistic spacetime representing astrophysical scenarios with black holes and neutron stars. I discuss single black hole spacetimes in Section 4.1, BBH initial data slices in Section 4.2, single neutron stars in Section 4.3, and a preview of BNS initial data slices in Section 4.4.

4.1 Single black hole spacetimes

Numerical solutions of the XCTS equations that represent Kerr spacetime are essential test cases for non-conformally-flat elliptic solves. I have already presented the convergence of the DG discretization error for a Schwarzschild black hole in Kerr-Schild coordinates in Section 2.4.3, and a detailed study of the elliptic solver convergence and parallel scaling for this problem in Section 3.4.2. These two studies dealt with a numerical subtlety that I detail in the following section. I also study the convergence of spinning Kerr solutions with negative-expansion boundary conditions, find apparent horizons in the data slices, and compute horizon quantities.

4.1.1 A numerical subtlety with apparent-horizon boundary conditions

The Newton-Raphon algorithm exhibits a divergence that occurs when apparent-horizon boundary conditions are coupled with the naive initial guess

$$\psi_0 = 1, \quad \alpha_0 = 1, \quad \beta_0^i = 0. \tag{4.1}$$

This defect also occurs in `SpEC` and I do not expect it to play a role in BBH initial data problems where we choose a smarter initial guess. However, the defect is sufficiently prominent in the commonly used XCTS formalism that it warrants a closer study. Specifically, it occurs with the configuration detailed in Section 2.4.3 that solves the XCTS equations for a Schwarzschild black hole in Kerr-Schild coordinates. The conformal metric is set to the spatial Kerr metric so the solution for the Kerr-Schild conformal factor is trivially $\psi = 1$, Eq. (3.37a), coinciding with the initial guess. This configuration, though reasonably straightforward, fails to converge when apparent-horizon boundary conditions (2.54) are imposed along with the Neumann-type lapse boundary condition

$$n^k \partial_k (\alpha\psi) = 0, \tag{4.2}$$

which is commonly employed in the literature.[1] The strongly nonlinear apparent-horizon boundary conditions couple the variables in such a way that the solution is initially corrected away from $\psi = 1$ and is then unable to recover. Note that the Neumann-type lapse boundary condition (4.2) yields a slice of Schwarzschild spacetime that differs slightly from the Kerr-Schild slice, Eq. (3.37). Specifically, for the configuration studied here the solution is $\psi = 1$ within a few percent.

Fig. 4.1 shows the numerical solution after the first Newton-Raphson step. It exhibits a deterioration from the initial guess away from the solution for the shift. The linearized apparent-horizon boundary conditions flip the shift vector inward in the vicinity of the inner shell boundary, where it should point outward. The reason for this effect is the linearization of Eq. (2.54), in particular

$$\delta\beta^i = \left(3\frac{(\alpha\psi)}{\psi^4}\delta\psi - \frac{\delta(\alpha\psi)}{\psi^3}\right) n^i, \tag{4.3}$$

which favors a correction of the shift aligned with the normal n^i (pointing out of the computational domain, i.e., toward the center of the shell) when $\delta\psi > 0$ and $\delta(\alpha\psi) < 0$. In the configuration considered here, both conditions are initially true. Consequently, the first Newton-Raphson step drives the numerical solution to a configuration from which it is unable to recover. I have confirmed this defect with the `SpEC` elliptic solver [55, 139], which diverges with the same behavior for this configuration.

The defect is less pronounced when a Dirichlet-type boundary condition is employed for the conformal factor instead of Eq. (4.2). Section 2.4.3 makes this choice, imposing the analytic Kerr-Schild lapse, Eq. (2.53b), at the inner shell surface so the exact Kerr-Schild slice is recovered. Many contemporary approaches to black hole initial data choose variations of Dirichlet-type lapse boundary conditions, such as most recent studies with the `SpEC` code [75, 78], and also Grandclément [133], Eq. (56).

1: See, e.g., Eq. (59a) in the early article by Cook and Pfeiffer [113], Eq. (44) in Lovelace et al. [75], and also contemporary studies such as Eq. (49) in Rashti et al. [108] or Eq. (61) in Papenfort et al. [89]. These studies simplify the problem by employing conformal flatness and/or maximal slicing, so they do not encounter the particular issue reported here.

[75]: Lovelace et al. (2008), *Binary-black-hole initial data with nearly-extremal spins*
[89]: Papenfort et al. (2021), *New public code for initial data of unequal-mass, spinning compact-object binaries*
[108]: Rashti et al. (2021), *Elliptica: a new pseudo-spectral code for the construction of initial data*
[113]: Cook and Pfeiffer (2004), *Excision boundary conditions for black hole initial data*

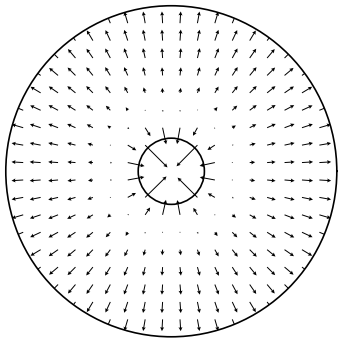


Figure 4.1: Numerical solution for the shift β^i in a slice through the shell-shaped computational domain after the first Newton-Raphson step with apparent-horizon boundary conditions and the Neumann-type lapse boundary condition, Eq. (4.2). For details on the configuration refer to Section 2.4.3 and input file `KerrSchildDefect.yaml`.

[75]: Lovelace et al. (2008), *Binary-black-hole initial data with nearly-extremal spins*
[78]: Varma, Scheel, and Pfeiffer (2018), *Comparison of binary black hole initial data sets*
[133]: Grandclément (2010), *KADATH: A spectral solver for theoretical physics*

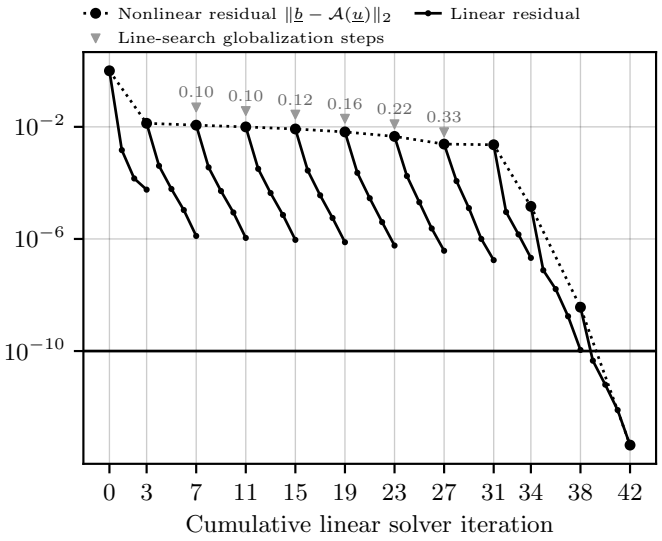


Figure 4.2: Convergence of the elliptic solver for the single black hole problem detailed in Section 2.4.3, i.e., with Dirichlet-type lapse boundary condition, from the naive initial guess $\psi_0 = 1$, $\alpha_0 = 1$, $\beta^i_0 = 0$, Eq. (4.1). The Newton-Raphson algorithm converges in eleven iterations, of which six require a line-search globalization step that reduces the step length λ to the fraction printed above the marker representing the full-step residual. Solved here is the configuration with $L = 1$ and $P = 5$ depicted in Fig. 2.10, circled in Fig. 2.11, and detailed in input file `KerrSchildDefect.yaml`.

Fig. 4.2 shows the convergence of the elliptic solver when the Dirichlet-type lapse boundary condition is chosen. The configuration is identical to Section 2.4.3 except for the initial guess, which is still Eq. (4.1).[2] The Newton-Raphson algorithm overshoots but converges successfully, albeit slowly, with the help of the line-search globalization described in Section 3.3.1. Evidently, the nonlinear apparent-horizon boundary conditions pose a hard problem for the Newton-Raphson algorithm but the elliptic solver proves capable of it. The solve presented in Fig. 4.2 completed in around ten seconds on 30 compute cores of the type detailed in Section 3.4.

2: Input file `KerrSchildDefect.yaml` (Appendix B.1)

The defect can be fully resolved with an initial guess that is closer to the solution for the shift, Eq. (2.53c). In BBH initial data problems we typically choose an initial guess derived from a superposition of Kerr solutions, which has proven sufficient to mitigate this problem. Another possible resolution that allows the single black hole problem to converge from the trivial initial guess, Eq. (4.1), is to choose a different conformal background $\bar{\gamma}_{ij}$ than the Kerr-Schild solution. For instance, I found the Newton-Raphson algorithm converges with the Neumann boundary condition (4.2) from the trivial initial guess (4.1) when $\bar{\gamma}_{ij} = \gamma^{\text{Kerr}}_{ij}/(1 + M/(2r))^4$, so the solution for the conformal factor is $\psi = 1 + M/(2r)$ (resembling isotropic coordinates) instead of $\psi = 1$.

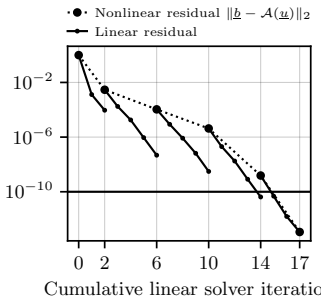


Figure 4.3: Replication of the bottom panel of Fig. 3.16, for comparison with Fig. 4.2. The only difference between the two studies is the choice of apparent-horizon boundary conditions with a Dirichlet lapse condition in Fig. 4.2, and linear Dirichlet and Neumann boundary conditions derived from the analytic Kerr-Schild solution here. The remaining DG discretization error (not replicated here) is approximately the same between the two studies at about 10^{-6}.

The preceding chapters dealt with this defect in different ways. In Section 2.4.3 we chose an initial guess close to the analytic solution in order to study the DG discretization error with apparent-horizon boundary conditions. Since the DG discretization error is independent of the convergence behavior of the Newton-Raphson algorithm (see the discussion in Section 3.4.1 related to Fig. 3.13) we initialized the elliptic solver with the analytic solution so it converges quickly toward the numerical solution. On the other hand, in Section 3.4.2 we begin at a flat initial guess to study the convergence behavior of the elliptic solver. We choose a combination of Dirichlet and Neumann boundary conditions that mimic apparent-horizon boundary conditions but remain linear. The convergence of the Newton-Raphson algorithm for this choice is presented in Fig. 3.16, and replicated here in Fig. 4.3 to the side.

It can be contrasted with Fig. 4.2, which differs only in the choice of inner boundary conditions. With the additional insight gained in this section we could have chosen apparent-horizon boundary conditions and a Dirichlet-type lapse condition in Section 3.4.2 just as well. The convergence of the Newton-Raphson algorithm has no effect on the results presented in Section 3.4.2, which concern the convergence, scaling and parallel performance of the preconditioned Krylov solver. Finally, in the BBH problems studied in Section 2.4.4 and Section 3.4.3 we employ apparent-horizon boundary conditions. The defect is resolved because we choose a superposition of Kerr solutions as initial guess. I present convergence studies of the Newton-Raphson algorithm for BBH problems further down in this chapter.

4.1.2 Spins and apparent horizons

Previous chapters presented only nonspinning black hole solutions. Here, I generalize these results and study spins with deformed excision surfaces, negative-expansion boundary conditions, and apparent-horizon quantities. To this end, I employ the spinning apparent horizon boundary conditions introduced in Section 1.2.3, Eq. (1.59), that involve rotation parameters $\boldsymbol{\Omega}_r$.

The complication that arises from spinning apparent-horizon boundary conditions is related to the choice of the excision surface. The boundary conditions *impose* that the excision surface is a spinning apparent horizon, so the slice of spacetime depends on our choice for the placement and shape of the boundary. The excision surface must be chosen consistently with the free data, such as the conformal metric $\bar{\gamma}_{ij}$, to allow for equilibrium solutions.

For example, a coordinate sphere cannot represent the horizon of a spinning Kerr solution in Kerr-Schild coordinates. The inner and outer Kerr horizons are located at constant Boyer-Lindquist radii [3]

$$r_\pm = M + \sqrt{M^2 - \boldsymbol{a}^2}, \tag{4.4}$$

which describe ellipsoids in Kerr-Schild coordinates (see Fig. 4.4). These ellipsoids of constant Boyer-Lindquist radius $r_{\rm BL}$ are defined by [4]

$$\boldsymbol{x}^2 = r_{\rm BL}^2 + \boldsymbol{a}^2 - \frac{(\boldsymbol{a}\cdot\boldsymbol{x})^2}{r_{\rm BL}^2} \tag{4.5a}$$

$$= r_{\rm BL}^2 \frac{r_{\rm BL}^2 + \boldsymbol{a}^2}{r_{\rm BL}^2 + (\boldsymbol{a}\cdot\hat{\boldsymbol{x}})^2} \tag{4.5b}$$

or, inversely,

$$r_{\rm BL}^2 = \frac{1}{2}\left(\boldsymbol{x}^2 - \boldsymbol{a}^2 + \sqrt{(\boldsymbol{x}^2 - \boldsymbol{a}^2)^2 + 4\,(\boldsymbol{a}\cdot\boldsymbol{x})^2}\right) \tag{4.5c}$$

in Kerr-Schild coordinates $\boldsymbol{x}$. If we were to excise a coordinate sphere from a spinning Kerr-Schild conformal background and impose apparent-horizon boundary conditions, the solution would be a perturbed black

3: Eq. (6.82) in Carroll [15]

4: Eq. (47) in Visser [215]

[215]: Visser (2007), *The Kerr spacetime: A Brief introduction*

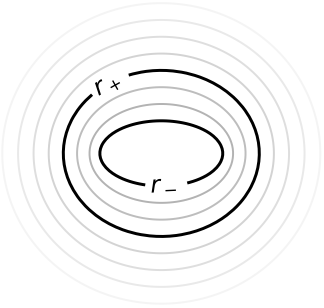


Figure 4.4: Surfaces of constant Boyer-Lindquist radius $r_{\rm BL}$ in Kerr-Schild coordinates, Eq. (4.5), for dimensionless spin $\chi = a/M = 0.9$. The spin is aligned with the vertical axis (side view).

hole. Therefore, we deform the boundary at which we impose apparent-horizon boundary conditions so it conforms to surfaces of constant Boyer-Lindquist radius r_{BL}.

Horizon-conforming excision surfaces

To deform the inner boundary of a spherical shell but keep the outer boundary spherical at a finite radius, I taper off the deformation toward the outer boundary. When the boundaries of the shell are spherical in coordinates $\mathring{x}^i$, I apply a "shape" coordinate map

$$x^i = \mathring{x}^i - (\mathring{x}^i - \mathring{x}^i_{\mathrm{C}})\, f(\mathring{r})\, \lambda(\theta, \phi), \tag{4.6}$$

where $\lambda(\theta, \phi)$ describes a radial distortion, $f(\mathring{r})$ is a transition function responsible for tapering off the deformation, and $\mathring{\boldsymbol{x}}_{\mathrm{C}}$ is the center of the radial distortion. The transition function for the spherical shell with inner radius $\mathring{r}_{\mathrm{min}}$ and outer radius $\mathring{r}_{\mathrm{max}}$ is

$$f(\mathring{r}) = \frac{\mathring{r}_{\mathrm{min}}\mathring{r}_{\mathrm{max}}}{\mathring{r}_{\mathrm{max}} - \mathring{r}_{\mathrm{min}}} \frac{1}{\mathring{r}} - \frac{\mathring{r}_{\mathrm{min}}}{\mathring{r}_{\mathrm{max}} - \mathring{r}_{\mathrm{max}}}, \tag{4.7}$$

so $f(\mathring{r}_{\mathrm{min}}) = 1$ and $f(\mathring{r}_{\mathrm{max}}) = 0$. This means a coordinate sphere of radius $\mathring{r} \in [\mathring{r}_{\mathrm{min}}, \mathring{r}_{\mathrm{max}}]$ around $\mathring{\boldsymbol{x}}_{\mathrm{C}}$ is mapped to a distorted surface with radius

$$r(\mathring{r}, \theta, \phi) = \mathring{r}\, \big(1 - f(\mathring{r})\, \lambda(\theta, \phi)\big) \tag{4.8}$$

in the coordinates x^i. At the inner boundary of the shell the deformation is

$$\lambda(\theta, \phi) = 1 - \frac{r_{\mathrm{min}}(\theta, \phi)}{\mathring{r}_{\mathrm{min}}}, \tag{4.9}$$

where $r_{\mathrm{min}}(\theta, \phi) > 0$ is freely specifiable and defines the deformation.

I choose $r_{\mathrm{min}}(\theta, \phi)$ so the inner boundary has constant Boyer-Lindquist radius $\mathring{r}_{\mathrm{min}}$,

$$r_{\mathrm{min}}(\theta, \phi) = \mathring{r}_{\mathrm{min}} \sqrt{\frac{\mathring{r}^2_{\mathrm{min}} + \boldsymbol{a}^2}{\mathring{r}^2_{\mathrm{min}} + (\boldsymbol{a} \cdot \hat{\boldsymbol{x}})^2}}, \tag{4.10}$$

using Eq. (4.5b), where the angular dependence is encoded in $\hat{\boldsymbol{x}} = (\cos\phi \sin\theta, \sin\phi \sin\theta, \cos\theta)$. Now the inner boundary of the shell conforms to the outer Kerr horizon in Kerr-Schild coordinates when we choose $\mathring{r}_{\mathrm{min}} = r_+$, Eq. (4.4), or to any other Boyer-Lindquist radius r_{BL} by another choice of $\mathring{r}_{\mathrm{min}} = r_{\mathrm{BL}}$. The right panel of Fig. 4.5 illustrates the horizon-conforming deformation of a spherical shell.

The transition to a spherical outer boundary of the shell also allows composing a domain for binary black holes, such as depicted in Fig. 3.19. The strategy to conform the initial excision boundary to a Kerr-Schild horizon has been employed in the `SpEC` code since Ref. [75]. However, the `SpEC` initial data solver does not transition to a spherical outer boundary because it employs overlapping subdomains that do not need to attach seamlessly. The configuration detailed here comes closer to the evolution domains in `SpEC`, where a shape map of the form (4.6) is employed to deform excision surfaces dynamically during the evolution.

[75]: Lovelace et al. (2008), *Binary-black-hole initial data with nearly-extremal spins*

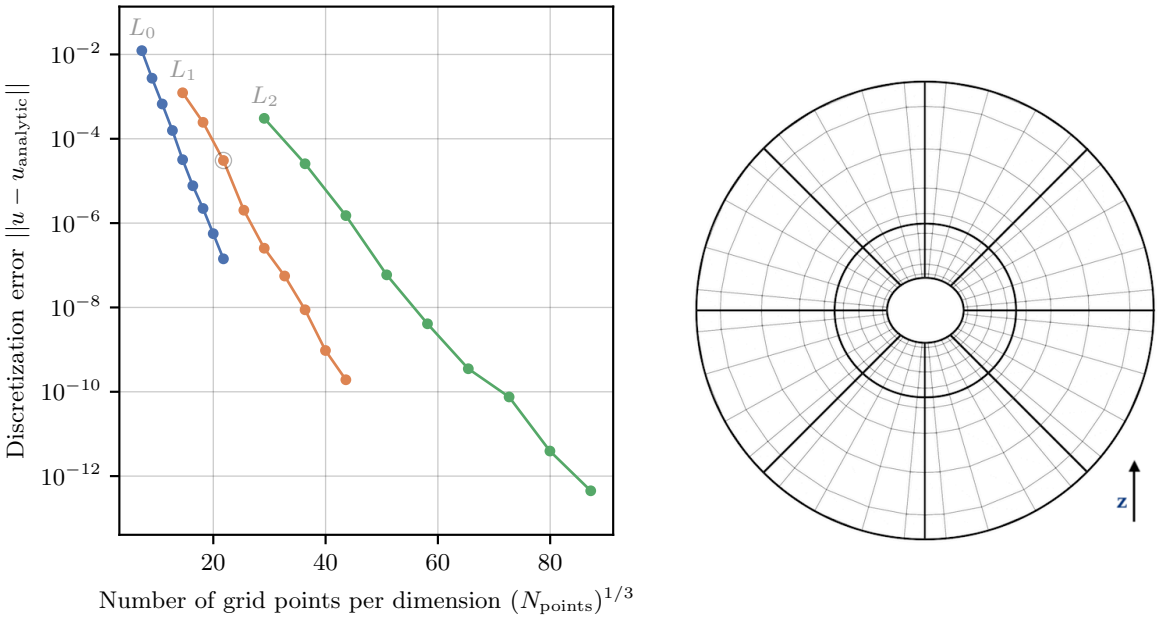


Figure 4.5: *Left:* Convergence of the DG discretization error with spinning apparent-horizon boundary conditions, Eq. (1.59), for a Kerr-Schild black hole with $\boldsymbol{\chi} = (0, 0, 0.9)$. Each of the three h-refinement levels span configurations from four to twelve grid points per element and dimension (polynomial order $P = 3$ to $P = 11$). The discretization error to the analytic Kerr-Schild solution is defined by Eq. (2.46). *Right:* A slice through the three-dimensional spherical shell domain with a horizon-conforming inner boundary used in these simulations. Pictured is the configuration with $L = 1$ and $P = 5$ that is circled in the convergence plot on the left.

Spinning Kerr-Schild solution

As a first test, I now show numerically that the apparent-horizon boundary conditions with spin, Eq. (1.59), are consistent with a Kerr black hole in Kerr-Schild coordinates. To this end I excise an ellipsoid with constant Boyer-Lindquist radius r_+ and choose the rotation parameters [5]

$$\boldsymbol{\Omega}_r = \frac{\boldsymbol{\chi}}{2r_+}, \tag{4.11}$$

so they coincide with the horizon angular velocity of a Kerr solution with dimensionless spin $\boldsymbol{\chi} = \boldsymbol{a}/M$. I solve the XCTS equations as detailed in Section 2.4.3. The spinning Kerr-Schild solution is [215]

$$\alpha = (1 + 2H)^{-1/2}, \tag{4.12a}$$
$$\beta^i = 2H\alpha^2 l^i, \tag{4.12b}$$

with the background quantities

$$\bar{\gamma}_{ij} = \delta_{ij} + 2Hl_i l_j, \tag{4.12c}$$
$$K_{ij} = 2\alpha\Big(H\partial_{(i}l_{j)} + l_{(i}\partial_{j)}H + 2H^2 l_{(i}l^k\partial_k l_{j)} + Hl_i l_j l^k \partial_k H\Big), \tag{4.12d}$$
$$K = \gamma^{ij}K_{ij}, \tag{4.12e}$$

5: See, e.g., Eq. (6.92) in Carroll [15] and note that $r_+^2 + \boldsymbol{a}^2 = 2Mr_+$.

[215]: Visser (2007), *The Kerr spacetime: A Brief introduction*

where

$$H = \frac{M r_{\mathrm{BL}}^3}{r_{\mathrm{BL}}^4 + (\boldsymbol{a}\cdot\boldsymbol{x})^2}, \tag{4.12f}$$

$$\boldsymbol{l} = \frac{r_{\mathrm{BL}}\boldsymbol{x} - \boldsymbol{a}\times\boldsymbol{x} + (\boldsymbol{a}\cdot\boldsymbol{x})\,\boldsymbol{a}/r_{\mathrm{BL}}}{r_{\mathrm{BL}}^2 + \boldsymbol{a}^2}, \tag{4.12g}$$

and where r_{BL} is the Boyer-Lindquist radius defined in Eq. (4.5). Note that Eq. (4.12) reduces to Eq. (2.53) for $\boldsymbol{a} = 0$. The choice of the Kerr-Schild spatial metric for the conformal metric, Eq. (4.12c). implies that the solution for the conformal factor is $\psi = 1$.

The left panel of Figure 4.5 shows the convergence of the DG discretization error to the analytic Kerr-Schild solution with dimensionless spin $\boldsymbol{\chi} = (0, 0, 0.9)$. The domain configuration is the same as detailed in Section 2.4.3 but with the new horizon-conforming inner boundary, as pictured in the right panel of Fig. 4.5.[6] The convergence is exponential with a rate slightly lower than that for the nonspinning case presented in Fig. 2.11.

6: Input file `KerrSchildSpin.yaml` (Appendix B.2)

Evidently, the Kerr-Schild slices, Eq. (4.12), fulfill the spinning apparent-horizon boundary boundary conditions, Eq. (1.59), with rotation parameters (4.11) at the ellipsoidal surface of constant Boyer-Lindquist radius r_+. The elliptic solver is equipped to reproduce this solution. A useful exercise for future explorations of spinning Kerr slices in coordinates different than Kerr-Schild, such as harmonic or spherical Kerr-Schild coordinates, is to check their consistency with Eq. (1.59).

4.1.3 Negative-expansion boundary conditions

Evolutions of binary black holes with the generalized harmonic formalism proceed on slices that extend slightly into the apparent horizons, so the dynamic horizon finder can always track the horizons. Therefore, initial data must be available also somewhat inside the apparent horizons. For a long time, the `SpEC` code has accomplished this by solving initial data with apparent-horizon boundary conditions and then extrapolating the initial data into the horizons.[7] More recently, Varma, Scheel, and Pfeiffer [78] have proposed to replace this procedure by *negative-expansion boundary conditions* that require no extrapolation. The idea is to impose the expansion on the boundary to be slightly negative, instead of zero, so it represents a surface *within* the apparent horizon. We accomplish this by adding the quantity

$$\frac{\psi^3}{4}\Theta_{\mathrm{Kerr}} \tag{4.13a}$$

to the Neumann-type boundary condition for the conformal factor, Eq. (1.59a), and the quantity

$$\epsilon = s_i\beta^i_{\mathrm{Kerr}} - \alpha_{\mathrm{Kerr}} \tag{4.13b}$$

to the orthogonal part $s_i\beta^i$ of the Dirichlet-type boundary condition for the shift, Eq. (1.59b).[8] Here, the quantities Θ_{Kerr}, α_{Kerr}, and β_{Kerr} are the expansion, lapse, and shift of a Kerr solution evaluated at the boundary, where the expansion Θ is given by Eq. (1.58). They are axisymmetric, but may vary along the azimuthal direction. Note that both corrections,

7: See Pfeiffer [170], Section 7.9.

[170]: Pfeiffer (2003), *Initial Data for Black Hole Evolutions*

[78]: Varma, Scheel, and Pfeiffer (2018), *Comparison of binary black hole initial data sets*

8: See Eqs. (25) and (26) in Varma, Scheel, and Pfeiffer [78].

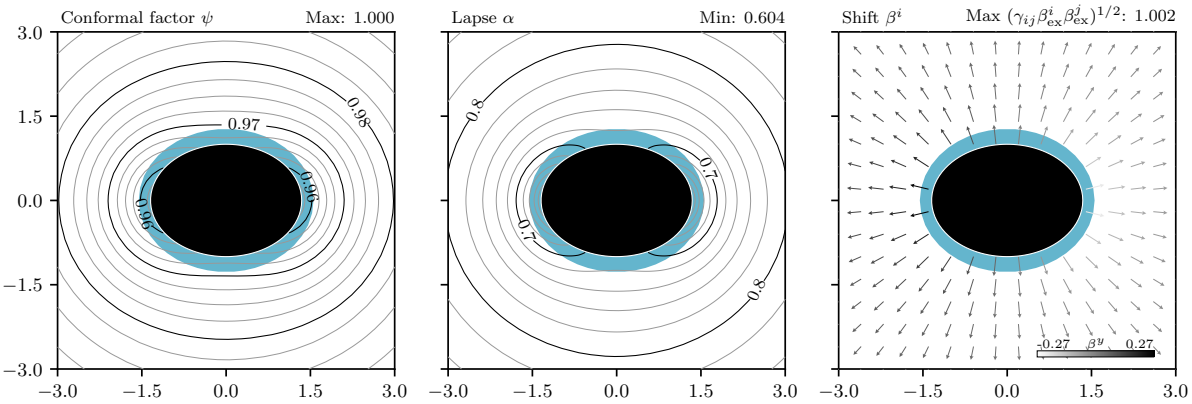


Figure 4.6: Kerr-Schild black hole with negative-expansion boundary conditions. The black region is excised from the computational domain and the boundary conditions are imposed on its surface. The blue region lies inside the (numerically determined) apparent horizon, but is part of the computational domain.

Eqs. (4.13a) and (4.13b), vanish when the boundary is a horizon of the solution.

Figure 4.6 presents a solution to the XCTS equations representing a spinning Kerr-Schild black hole with negative-expansion boundary conditions. It differs from the Kerr-Schild configurations solved in Fig. 4.5 only in the choice of inner boundary, and the resulting corrections to the apparent-horizon boundary conditions that follow from the choice of boundary away from the horizon, Eq. (4.13).[9] Specifically, I excise an ellipsoid of constant Boyer-Lindquist radius $\mathring{r}_{\rm min} = M$, which is halfway between r_+ and r_-. To obtain a dimensionless spin of approximately χ I choose

$$\boldsymbol{\Omega}_r = \frac{\chi}{2\mathring{r}_{\rm min}}, \tag{4.14}$$

following Eq. (4.11).

9: Input file `KerrSchildSpin.yaml` (Appendix B.2)

The left panel in Fig. 4.7 shows the convergence of the discretization error with hp refinement for the spinning Kerr-Schild black hole with negative-expansion boundary conditions. Since this configuration only approximates a Kerr-Schild solution, I compute the discretization error as detailed in Section 2.4.4, by interpolating the solution variables to a set of sample points $\boldsymbol{x}_m$ and computing the L_2 norm of the difference to a high-resolution reference run, Eq. (2.59). I choose the sample points $\boldsymbol{x}_1 = (1.6, 0, 0)$ (near the bulge of the horizon), $\boldsymbol{x}_2 = (0, 0, 1.2)$ (near the top of the horizon), and $\boldsymbol{x}_3 = (9, 0, 0)$ (outside the horizon). The discretization error $\|u - u_{\rm ref}\|$ converges exponentially. Also shown is the exponential convergence of the constraint norm discussed in Section 4.2.4 below. All simulations, which involve up to ~ 26.5 million degrees of freedom for the $L = 3$ and $P = 11$ reference run, completed in at most a few minutes on up to 240 cores.

Apparent horizon quantities

Since the inner boundary now extends into the apparent horizon, we can employ an apparent horizon finder to determine the location of the apparent horizon on the slice numerically. I use the apparent horizon finder implemented in the `SpECTRE` code, which is based on the parabolic

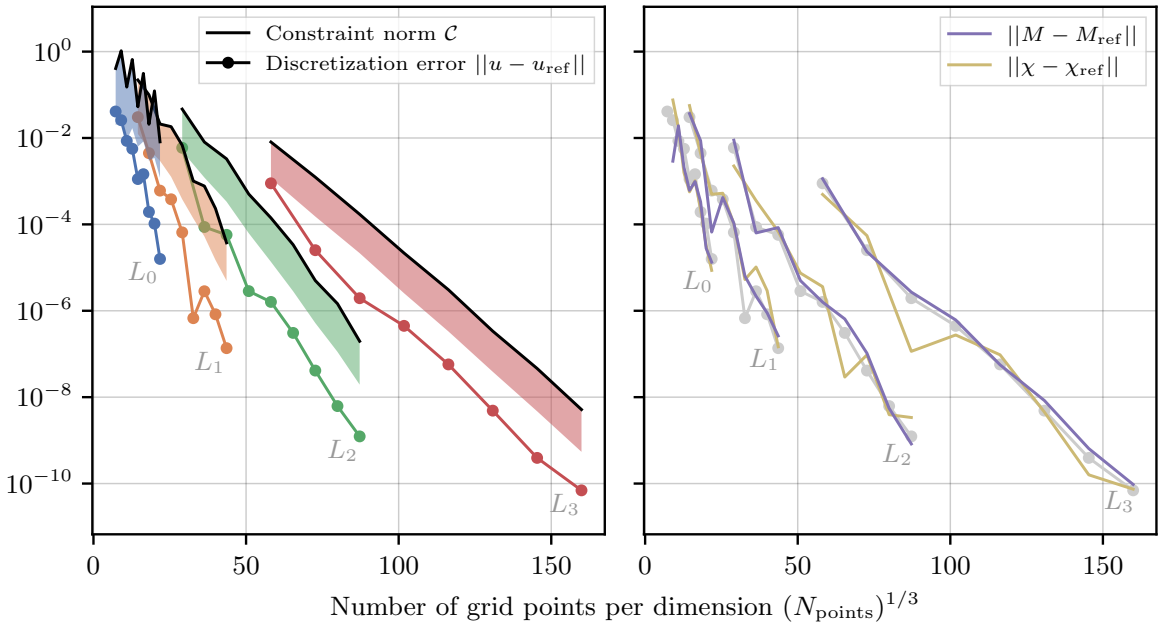


Figure 4.7: *Left:* Convergence of the Kerr-Schild problem with negative-expansion boundary conditions, at h-refinement levels $L \in [0, 3]$ and $P \in [3, 11]$. The constraint norm $\mathcal{C}$ and the shaded area below are defined in Section 4.2.4. The discretization error is measured to the highest-resolution reference configuration ($L = 3$, $P = 11$). *Right:* Convergence of apparent horizon quantities. The light gray lines illustrate the discretization error, as shown in the left panel.

relaxation method by Gundlach [115]. The numerically determined apparent horizon is shaded blue in Fig. 4.6. On it we can extract the irreducible mass M_{irr}, the spin magnitude S, the Christodoulou (or "Kerr") mass M, and the dimensionless spin χ, as

$$M_{\mathrm{irr}}^2 = \frac{A}{16\pi}, \tag{4.15}$$

$$S = \frac{1}{8\pi} \oint z\Omega \, \mathrm{d}A \,, \tag{4.16}$$

$$M^2 = M_{\mathrm{irr}}^2 + \frac{S^2}{4M_{\mathrm{irr}}^2}, \tag{4.17}$$

$$\chi = \frac{S}{M^2}, \tag{4.18}$$

where A is the area of the apparent horizon,[10] and S is the quasilocal spin magnitude measured by solving a generalized eigenvalue problem for the spin potential z and projecting it on the spin function Ω over the apparent horizon surface [75].[11]

The right panel in Fig. 4.7 shows the convergence of the Christodoulou mass M and the dimensionless spin magnitude χ measured on the apparent horizon. Their errors are computed as the difference to the highest-resolution reference run also used in the left panel. I find $M \approx 0.836$ and $\chi \approx 0.864$. These values deviate from the parameters $M = 1$ and $\chi = 0.9$ of the background Kerr-Schild solution defining the conformal metric and trace of the extrinsic curvature, because the negative-expansion boundary conditions are only approximately consistent with a Kerr-Schild slice. Instead, the slice solves the Einstein constraints but represents a perturbed Kerr solution in approximate Kerr-Schild coordinates. For

[115]: Gundlach (1998), *Pseudospectral apparent horizon finders: An Efficient new algorithm*

10: See, e.g., Eq. (7.2) in Baumgarte and Shapiro [17] for M_{irr}, and see Appendix D in Baumgarte et al. [216] or Appendix C in Baumgarte and Shapiro [17] for details on computing the area element on the apparent horizon.

[216]: Baumgarte et al. (1996), *Implementing an apparent horizon finder in three-dimensions*

11: See Eq. (9) in Ref. [217] for the definition of the spin potential z in terms of an eigenvalue problem, and Eq. (10) in Ref. [218] for the definition of the spin function Ω. We compute S as the Euclidean norm of Eq. (4.16) over the three spin potentials with the smallest eigenvalues.

[75]: Lovelace et al. (2008), *Binary-black-hole initial data with nearly-extremal spins*

[217]: Owen (2009), *The Final Remnant of Binary Black Hole Mergers: Multipolar Analysis*

[218]: Owen et al. (2019), *Black Hole Spin Axis in Numerical Relativity*

this reason $\psi \neq 1$ at the percent level in Fig. 4.6. In practice, we would perform an iteration procedure to find mass and rotation parameters so M and χ take the desired values, such as the procedure detailed by Ossokine et al. [181] used in SpEC. Nevertheless, a careful study with the goal to optimize the negative-horizon boundary conditions so the measured horizon quantities come as close as possible to the background Kerr-Schild solution has the potential to reduce perturbations in initial data slices.

[181]: Ossokine et al. (2015), *Improvements to the construction of binary black hole initial data*

4.2 Binary black holes

For binary black holes (BBH) initial data I choose the background quantities in the XCTS equations based on superposed isolated solutions [75], and impose quasiequilibrium apparent-horizon boundary conditions on excised regions of the domain to represent black holes [113]. This procedure is detailed in Section 2.4.4 for orbiting, but nonspinning, superposed Kerr-Schild (SKS) initial data.

[75]: Lovelace et al. (2008), *Binary-black-hole initial data with nearly-extremal spins*

[113]: Cook and Pfeiffer (2004), *Excision boundary conditions for black hole initial data*

My implementation in SpECTRE is based on the new task-based parallel elliptic solver developed in Chapter 3 and Ref. [2]. It also supports spins, unequal masses, an initial radial velocity, horizon-conforming excision surfaces (see Section 4.1.2), and negative-expansion boundary conditions (see Section 4.1.3). This feature set is already comparable to SpEC. In SpEC, separate rootfinding routines control the free input parameters [181], correct the center of mass, and control eccentricity [219–221]. These routines invoke the elliptic solver repeatedly, and will be straightforward to adapt to the new code.

[2]: **Vu** et al. (2022), *A scalable elliptic solver with task-based parallelism for the SpECTRE code*. Chapter 3 of this thesis.

[181]: Ossokine et al. (2015), *Improvements to the construction of binary black hole initial data*

[219]: Pfeiffer et al. (2007), *Reducing orbital eccentricity in binary black hole simulations*

[220]: Buonanno et al. (2011), *Reducing orbital eccentricity of precessing black-hole binaries*

[221]: Mroue and Pfeiffer (2012), *Precessing Binary Black Holes Simulations: Quasi-circular Initial Data*

The new elliptic solver algorithms allow to scale initial data problems to supercomputers, which is the most prominent advancement presented in this thesis over the SpEC elliptic solver [139]. The superior parallel performance is demonstrated in Section 3.4.3 and Fig. 3.20 for equal-mass and nonspinning SKS initial data. In this section I present optimizations, test cases, and results not yet included in Chapter 3.

[139]: Pfeiffer et al. (2003), *A multidomain spectral method for solving elliptic equations*

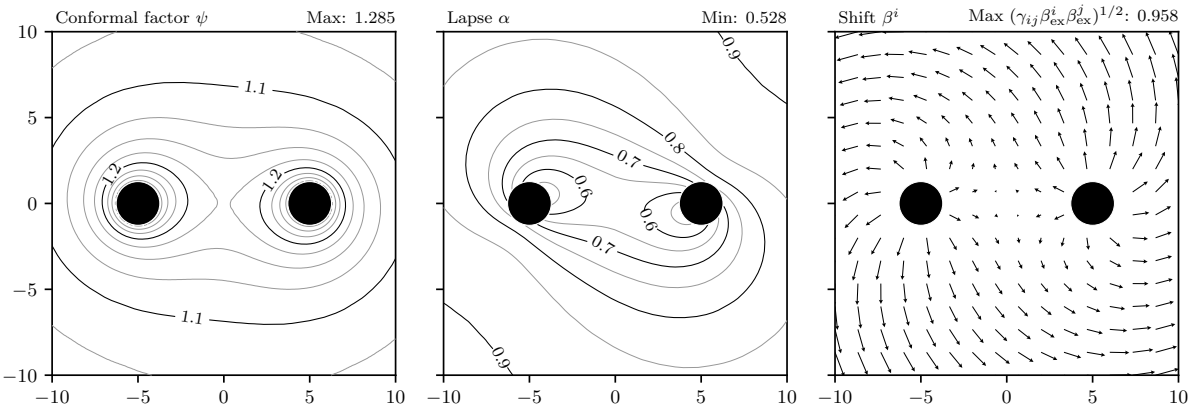


Figure 4.8: Binary black hole initial data in superposed isotropic Kerr-Schild coordinates. The background is conformally flat, $\bar{\gamma}_{ij} = \eta_{ij}$, the trace of the extrinsic curvature trace is the unmodulated superposition, $K = K_{\rm left} + K_{\rm right}$, the inner lapse boundary condition is of Neumann type, Eq. (4.2), and the outer boundary is at radius 10^6. This configuration reproduces Fig. 7.5 in Ref. [170].

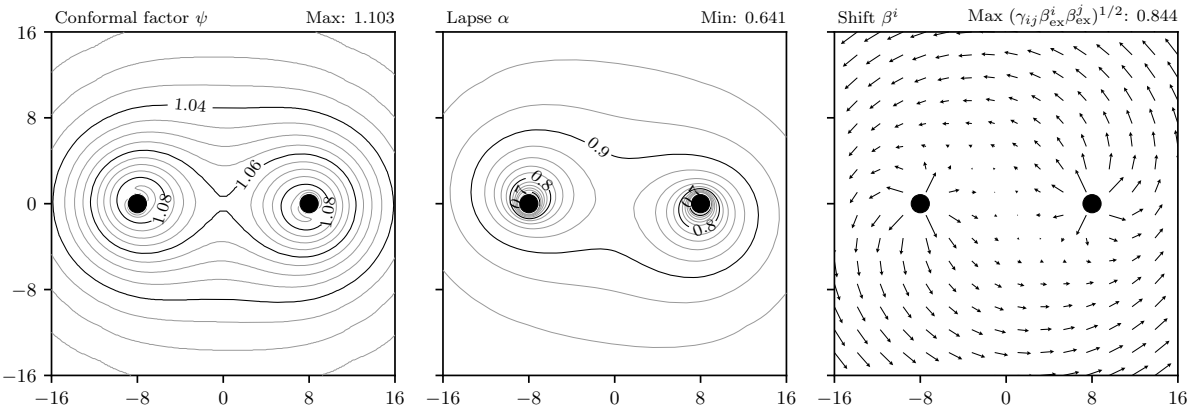


Figure 4.9: Binary black hole initial data in superposed Kerr-Schild coordinates solved in Section 3.4.3. The conformal background is a superposition of Kerr-Schild metrics modulated by Gaussians, the inner lapse boundary condition is of Dirichlet type, and the outer boundary is at radius 300 (see input file `BbhSks.yaml`).

Furthermore, my implementation in SpECTRE supports superposing any two isolated solutions in orbit, and is therefore neither limited to Kerr-Schild black holes, nor to vacuum solutions. At the time of writing, the following Schwarzschild solutions are implemented in addition to the spinning Kerr-Schild solution:

Isotropic Conformally flat, $\bar{\gamma}_{ij} = \eta_{ij}$, and maximally sliced, $K = 0$, coordinates for Schwarzschild spacetime that arise from the canonical Schwarzschild coordinates by a radial transformation.[12] This is traditionally a very convenient coordinate system for simple initial data problems, but generalizes poorly to spinning solutions because the Kerr spacetime does not admit conformally flat slices. The coordinates are singular at the horizon, but Cook and Pfeiffer [113] show that the lapse boundary condition at the horizon can select another maximal Schwarzschild slice that can be radially transformed to conformal flatness.[13]

Painlevé-Gullstrand In these coordinates the spatial metric is flat, $\gamma_{ij} = \eta_{ij}$, and the lapse is trivial, $\alpha = 1$.[14] In contrast to (isotropic) Schwarzschild coordinates the shift and extrinsic curvature are nontrivial in Painlevé-Gullstrand coordinates, making them useful to test the momentum sector of the Einstein constraint equations.

Isotropic Kerr-Schild (or Eddington-Finkelstein) Horizon penetrating and conformally flat coordinates that arise from the Schwarzschild metric in Kerr-Schild coordinates, Eq. (2.53), by a radial transformation. In contrast to isotropic Schwarzschild coordinates the radial transformation requires numerical rootfinding.[15]

Harmonic Horizon penetrating coordinates that are harmonic in both time and space,[16] meaning they satisfy the harmonic coordinate conditions $g^{\mu\nu}\nabla_\mu\nabla_\nu x^\rho = 0$.[17] Work is underway to generalize the harmonic Schwarzschild solution to include spin, which will enable superposed harmonic Kerr (SHK) initial data [78].

The matter solutions currently implemented in the SpECTRE initial data solver and available for superposition are listed in Section 4.4 below. A brief introduction to produce BBH initial data with SpECTRE is available online [223].

12: See Eq. (1.60) and Table 2.1 in Baumgarte and Shapiro [17].

[113]: Cook and Pfeiffer (2004), *Excision boundary conditions for black hole initial data*

13: See Sec. III.C in Cook and Pfeiffer [113], and Sec. 7.4.2 in Pfeiffer [170]. Note the missing factor C/r^4 in Eq. (7.44c) in Ref. [170].

[170]: Pfeiffer (2003), *Initial Data for Black Hole Evolutions*

14: See Table 2.1 in Baumgarte and Shapiro [17].

15: See Sec. 7.4.1 in Pfeiffer [170] for details.

16: See Eqs. (45) to (50) in Cook and Scheel [222] which represent the zero-spin limit of the time harmonic and horizon penetrating slices of Kerr spacetime presented in the paper. We add the radial transformation $r \to r + M$ to make the spatial coordinates harmonic as well (see Eq. (43) in Ref. [222]), so the coordinates remain harmonic under boosts.

[222]: Cook and Scheel (1997), *Well-behaved harmonic time slices of a charged, rotating, boosted black hole*

17: See Sec. 4.3 in Baumgarte and Shapiro [17] for details on harmonic coordinates. Note that Eq. (4.45) in Baumgarte and Shapiro [17] is missing a minus sign: $(\partial_t - \beta^j\partial_j)\beta^i = -\alpha^2(\gamma^{ij}\partial_j \ln\alpha - \gamma^{jk}\Gamma^i_{jk})$.

[78]: Varma, Scheel, and Pfeiffer (2018), *Comparison of binary black hole initial data sets*

[223]: **Vu** et al. (2022), *Binary black hole initial data example in the SpECTRE documentation*

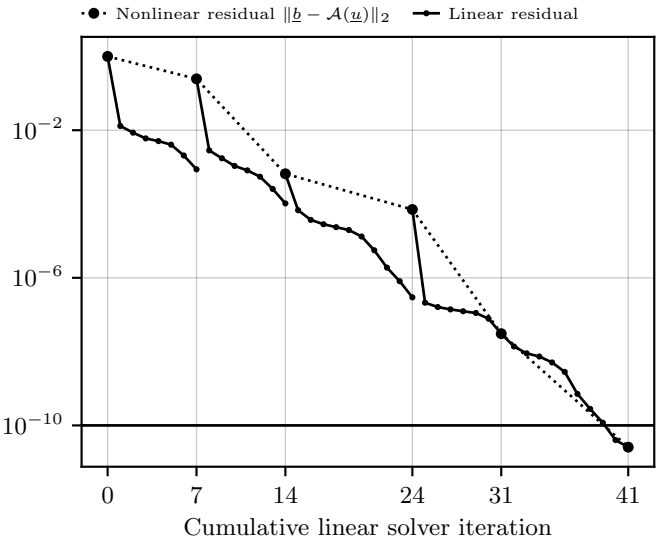


Figure 4.10: Convergence of the elliptic solver for the equal-mass and nonspinning SKS configuration marked ∗ in Fig. 3.20 and pictured in Fig. 3.19 and Fig. 4.9.

My implementation also allows to superpose two *different* isolated solutions to construct a conformal background, which has not yet been explored in the literature. For example, based on the conclusions drawn by Varma, Scheel, and Pfeiffer [78] the combination of Kerr-Schild coordinates for a highly spinning black hole, and harmonic coordinates for a second black hole with moderate spin, can be advantageous for evolutions.

[78]: Varma, Scheel, and Pfeiffer (2018), *Comparison of binary black hole initial data sets*

To connect to Pfeiffer [170], I present in Fig. 4.8 a BBH initial data slice in superposed isotropic Kerr-Schild coordinates with Neumann-type lapse boundary condition, Eq. (4.2), which reproduces Fig. 7.5 in Ref. [170].[18] This initial data slice is conformally flat.

[170]: Pfeiffer (2003), *Initial Data for Black Hole Evolutions*

18: Input file `BbhKsi.yaml` (Appendix B.3)

4.2.1 Superposed Kerr-Schild initial data

Superposed Kerr-Schild (SKS) initial data has seeded routine `SpEC` simulations for over a decade now [9, 75]. I have studied the applicability of the discontinuous Galerkin scheme to SKS initial data in Section 2.4.4, and the superior parallel performance of the new `SpECTRE` elliptic solver for the same problem in Section 3.4.3. Figure 4.9 shows a visualization of this equal-mass and nonspinning slice of SKS initial data.

[9]: SXS (2019), *The SXS collaboration catalog of binary black hole simulations*
[75]: Lovelace et al. (2008), *Binary-black-hole initial data with nearly-extremal spins*

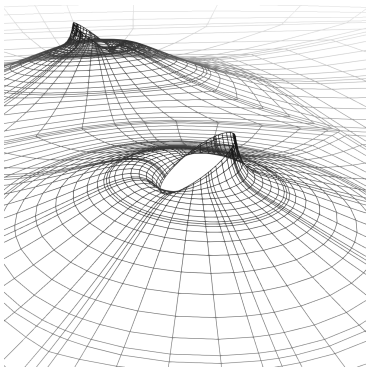


Figure 4.11: Deformation of the conformal factor near the horizons in SKS initial data, Fig. 4.9.

A notable feature in Fig. 4.9 is the deformation of the conformal factor close to the two black holes. It is depicted in more detail in Fig. 4.11. The feature also appears when this configuration is solved in `SpEC`, which is unsurprising since the difference of the configuration solved in `SpEC` and `SpECTRE` is convergent, as detailed in Section 3.4.3. The shape of the feature is indicative of an orbital effect. Since the conformal background is a superposition of isolated solutions at rest, the conformal factor accounts for both the gravitational interaction between the two bodies as well as the orbital motion. Therefore, it is likely that the feature diminishes when a Lorentz boost is added to the individual background solutions with a corresponding deformation of the excision surfaces to conform to the boosted isolated horizons. This has been the practice in `SpEC` since Lovelace et al. [75] but has never been studied in detail. Eliminating the

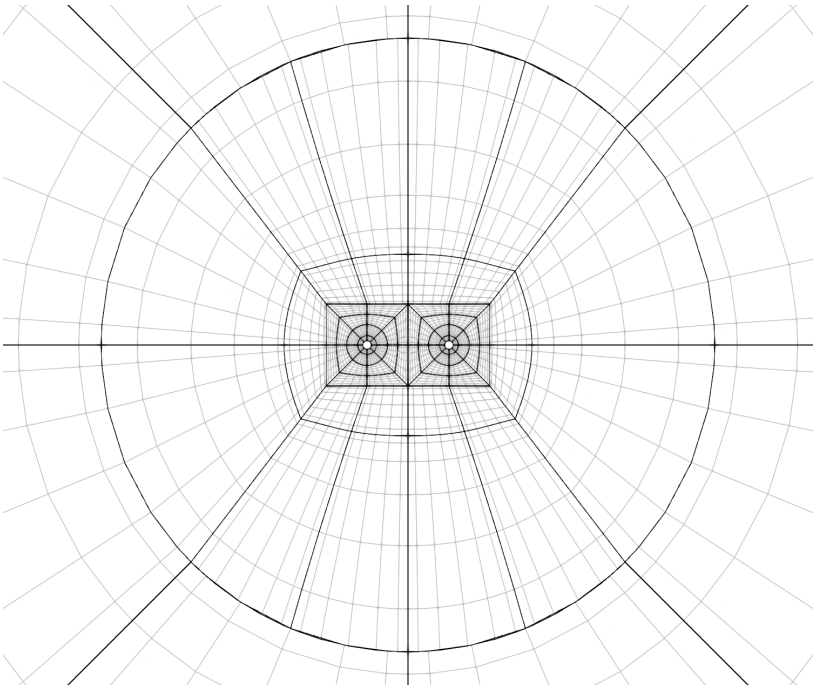


Figure 4.12: Slice through the optimized BBH domain with equiangular maps and a bulged envelope. The outer spherical shell is h refined such that wedges have equal angular size. All elements in this picture have seven angular grid points, and $\{7, 7, 8, 11\}$ radial grid points in the layers ordered from outermost to innermost.

sharp peaks visible in Fig. 4.11 potentially reduces the required resolution close to the excision surfaces, as well as nonequilibrium perturbations in the initial data slice.

Figure 4.10 shows the elliptic solver convergence for an SKS initial data run.[19] Specifically, it shows the run marked $*$ in Fig. 3.20, which corresponds to the domain pictured in Fig. 3.19 and replicated here in Fig. 4.13 to the side. It involves 54 irreducible blocks, hp refinement with nonconforming boundaries, and $\sim$ 700k grid points. Despite this significant increase in complexity from the simple spherical shells studied in, e.g., Fig. 3.16, the multigrid-Schwarz preconditioned GMRES algorithm drives the linear residual down by a factor of 10^{-3} in only seven to ten iterations per Newton-Raphson step. The solve completes in 44 seconds on 90 cores, and reaches an accuracy of $\lesssim 10^{-6}$ (see Fig. 3.20).

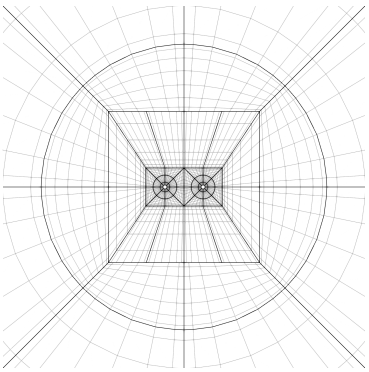


Figure 4.13: BBH domain (replicated from Fig. 3.19).

4.2.2 Domain optimizations

In an effort to accelerate the elliptic solver further, some of the domain optimizations suggested in Section 3.5 have been implemented in SpECTRE since the publication of Ref. [2]. Figure 4.12 illustrates the optimized domain.

[2]: **Vu** et al. (2022), *A scalable elliptic solver with task-based parallelism for the SpECTRE code*. Chapter 3 of this thesis.

Most prominently, we eliminated the layer of ten blocks that transitioned from the cube to the sphere enveloping the center in Fig. 4.13. This transition layer employed an inefficient linear radial coordinate distribution, and had to resolve the grid compression at the corners of the cube. Instead, we now bulge out the ten wedges of the cube to a sphere directly, bringing the number of blocks in the domain down to 44. The intermediate nonspherical layer shown in Fig. 4.12 arises from radial h refinement. This is preferable over block boundaries because it enables the multigrid solver to coarsen the grid by an additional level (see Section 3.3.3).

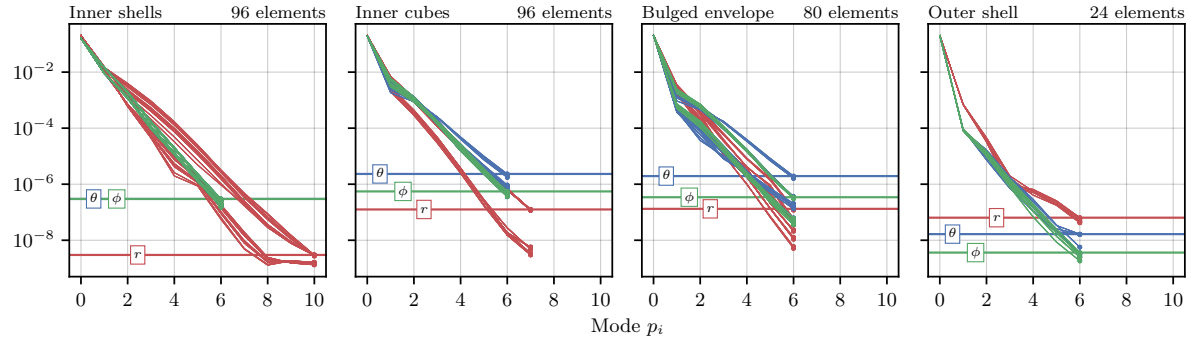


Figure 4.14: Power monitors for SKS initial data on the domain pictured in Fig. 4.12. Panels from left to right correspond to layers in the domain, from innermost (shells enclosing the excision surfaces) to outermost (spherical shell extending to the outer radius). Plotted is the power monitor (4.20) in every direction of every DG element in the domain. The five XCTS variables $u_A = \{\psi, \alpha\psi, \beta^i_{\text{excess}}\}$ are combined as an L_2 norm, so the plotted power monitor in direction i of element Ω_k is $(\sum_A P_{p_i}(\underline{u}_A)^2)^{1/2}$. Horizontal lines indicate the largest power in the highest mode over all elements in the panel, indicative of the truncation error of that axis.

Furthermore, a new equiangular coordinate map increases angular resolution considerably. It ensures that equally-spaced logical coordinates on a reference cube remain equally spaced in angular direction when the reference cube is deformed to a wedge (see Fig. 2.1b). The equiangular map is favorable for the spherical shells involved in the domain.

To inform domain optimizations I employ *power monitors*, which are diagnostic quantities geared toward an adaptive mesh refinement (AMR) algorithm.[20] First, I compute the modes $\tilde{u}_p$ associated with the nodal data u_p on an element Ω_k, as

$$\tilde{u}_p = \mathcal{V}^{-1}_{pq} u_q, \tag{4.19}$$

where $\mathcal{V}_{pq} = \prod_i^d \Phi_{q_i}(\xi_{p_i})$ is the *Vandermonde* matrix on the element. It transforms between the *nodal* representation, formulated in terms of Lagrange polynomials rooted at the LGL collocation points ξ_{p_i}, and the *modal* representation, formulated in terms of Legendre polynomials $\Phi_{p_i}(\xi)$.[21] Then, a definition of the power in the modes in dimension i of element Ω_k is

$$P_{p_i}(\underline{u}) = \sqrt{\frac{\sum_{\substack{q=1 \\ q_i=p_i}}^{N} |\tilde{u}_q|^2}{\sum_{j\neq i}^{d} N_j}}. \tag{4.20}$$

Recall from Section 2.3.1 that $p_i \in \{1, \dots, N_i\}$ enumerates the modes (or grid points) in dimension i of the element Ω_k, and $p \in \{1, \dots, N\}$ identifies the mode (or grid point) regardless of dimension (see Fig. 4.15 to the side, which replicates Fig. 2.1b). I have dropped the index k that identifies the element in these expressions.

Figure 4.14 shows the power monitors for SKS initial data on the domain pictured in Fig. 4.12. Modes decay exponentially, but at different rates. The magnitude of the highest mode (horizontal lines) gives an indication which direction of the grid dominates the discretization error. While this analysis has guided my domain optimizations, it may not yet be sufficient to inform an anisotropic AMR algorithm by itself. For example, I have found a high radial resolution in the inner shells reduces the discretization error considerably, but appears excessive in Fig. 4.14 (left panel).

20: See Eq. (52) in Szilágyi [224] for a definition of power monitors in the context of spherical harmonics employed in the SpEC code.

[224]: Szilágyi (2014), *Key Elements of Robustness in Binary Black Hole Evolutions using Spectral Methods*

21: See Section 2.3.1 for details on the domain decomposition, and Sec. 3.1 in Hesthaven and Warburton [126] for a discussion of the Vandermonde matrix.

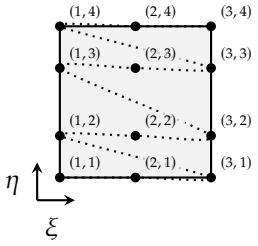


Figure 4.15: Enumeration of grid points on the regular grid of a two-dimensional element. The power monitor along direction i, Eq. (4.20), marginalizes over all orthogonal dimensions $j \neq i$ of the grid. This illustration is a replication of Fig. 2.1b.

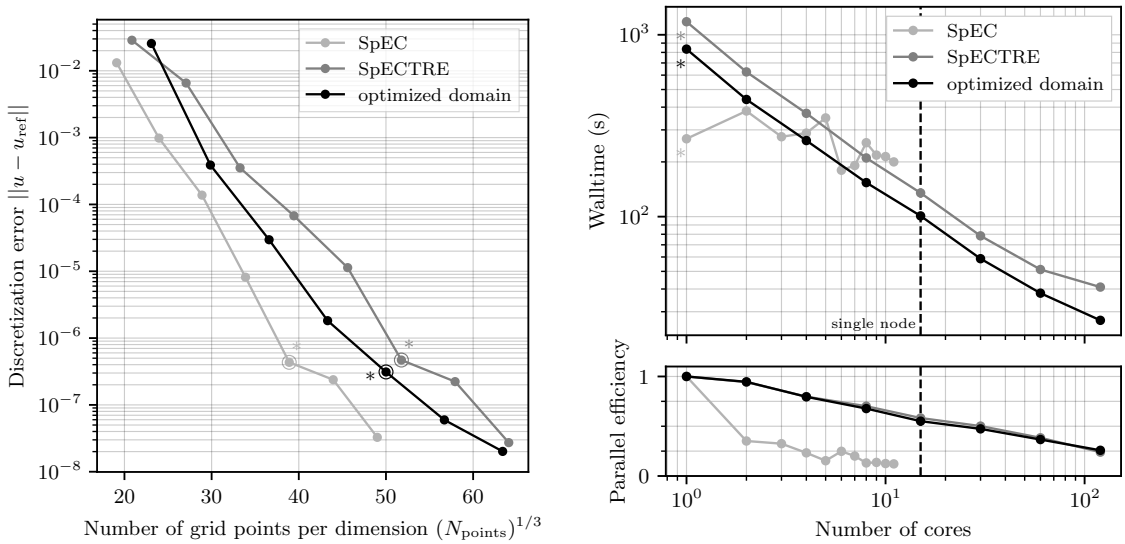


Figure 4.16: Improved convergence and speed accomplished by recent domain optimizations for the BBH initial data problem solved in Section 3.4.3. The gray lines correspond to the data shown in Fig. 3.20, and the black line corresponds to the optimized domain (Fig. 4.12 and input file `BbhDomain.yaml`). *Left:* The optimized domain requires fewer grid points to reach the same accuracy. *Right:* The run marked $*$ on the optimized domain is faster than `SpEC` on four cores. It scales reliably to 120 cores, at which point it is about ten times faster than `SpEC`.

Figure 4.16 presents an update to Fig. 3.20 with an additional curve representing the convergence of the DG discretization error on the optimized BBH domain.[22] The optimized domain achieves the same accuracy with fewer grid points, hence accelerating the elliptic solver. The run marked $*$ completes in 28 seconds on 90 cores while reaching an accuracy of $\lesssim 10^{-6}$, which is over 50 % faster than the previous configuration.

22: Input file `BbhDomain.yaml` (Appendix B.5)

Further domain optimizations are possible, some of which are already mentioned in Section 3.5. Work is underway to equalize the angular size of wedges, which is a flaw apparent in Fig. 4.12. Clearly, the wedges toward the left and right of the bulged envelope are larger than the wedges toward the top and bottom in Fig. 4.12. This flaw also manifests in h-nonconforming boundaries between the outer shell and the bulged envelope.

To illustrate this point, Fig. 4.17 shows all block boundaries in the domain. The angle ϕ is covered by four outer wedges of equal size. However, the angle θ is covered by six outer wedges: two full-size wedges plus four half-wedges split by the vertical symmetry plane. To obtain outer wedges of equal angular size, I first split all outer wedges in two along the angle ϕ. Then, I also split the two full-size outer wedges at the poles of θ in two along both angles. To carry this h refinement inward, I split all wedges of the bulged envelope in two along both angles to conform to the two central cubes. The resulting asymmetry in resolution along θ is evident in Fig. 4.12. It is also visible in the power monitors in Fig. 4.14, where the residual power in a few angular elements dominates the discretization error in the bulged envelope, and carries over into the inner cubes. This issue can be resolved by an equatorial compression such that all wedges cover 60° in θ and 90° in ϕ. The asymmetry between the two angles can be compensated by p refinement.

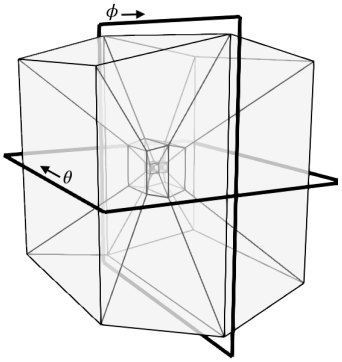


Figure 4.17: Block boundaries in the BBH domain.

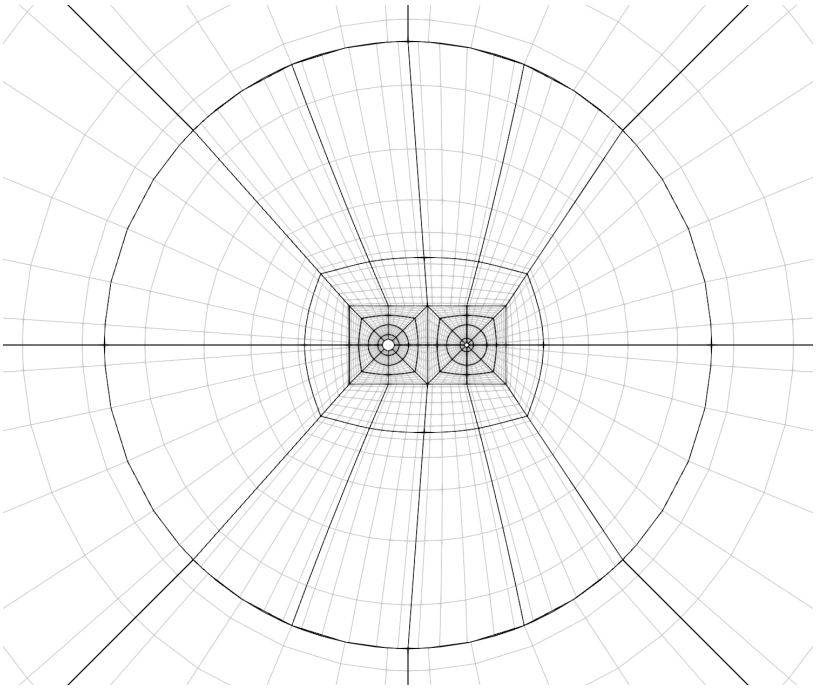


Figure 4.18: Slice through the BBH domain based on Fig. 4.12 with unequal masses and an ellipsoidal excision surface on the left. The shell around the excision surface transitions to a coordinate sphere.

4.2.3 Spinning and unequal-mass SKS initial data with negative-expansion boundary conditions

In this section I present an SKS initial data slice with spin, unequal masses, and negative-expansion boundary conditions. I find apparent horizons in the data and study the convergence of horizon quantities. For this study I choose the case labeled `q3` by Ossokine et al. [181], which has $q = 3$, $\chi_1 = (0, 0.49, -0.755)$, and $\chi_2 = (0, 0, 0)$.[23]

[181]: Ossokine et al. (2015), *Improvements to the construction of binary black hole initial data*

23: Input file `BbhSpin.yaml` (Appendix B.6)

The black holes are placed at coordinate separation $D_0 = 15.48$ and orbit with $\Omega_0 = 0.01515$ (see Table 1 in Ref. [181]). I approximate the center of mass by $C_{\mathrm{COM}} = (M_1 C_1 + M_2 C_2)/M$, so the left black hole with Kerr mass parameter $M_1 = 0.75$ is placed at $x = -3.87$, and the right black hole with $M_2 = 0.25$ is placed at $x = 11.61$. I choose falloff widths $w_1 = 7.5$ and $w_2 = 2.5$ for the superposition.

I excise ellipsoids of constant Boyer-Lindquist radius $\mathring{r}_{\mathrm{min}}$ as detailed in Section 4.1.2, choosing the spin of the respective Kerr-Schild solution to define the Boyer-Lindquist radius for each of the two excision surfaces, Eq. (4.10). For the left black hole I excise at $\mathring{r}_{\mathrm{min},1} = 0.97$, and for the right black hole at $\mathring{r}_{\mathrm{min},2} = 0.45$. These choices correspond to $\mathring{r}_{\mathrm{min}} \approx 0.9 r_+$, so the excision surfaces lie within the horizons of the two isolated solutions. I impose spinning apparent-horizon boundary conditions, Eq. (1.59), with negative-expansion corrections, Eq. (4.13), on the ellipsoidal excision surfaces, and choose the rotation parameters

$$\Omega_r = \frac{\chi}{2\mathring{r}_{\mathrm{min}}} \tag{4.21}$$

for each of the two excisions, following Eq. (4.14).

To solve the XCTS problem numerically I employ the optimized domain detailed in Section 4.2.2. It accounts for the shifted center of mass by distorting the bulged envelope, and for the ellipsoidal excision surface

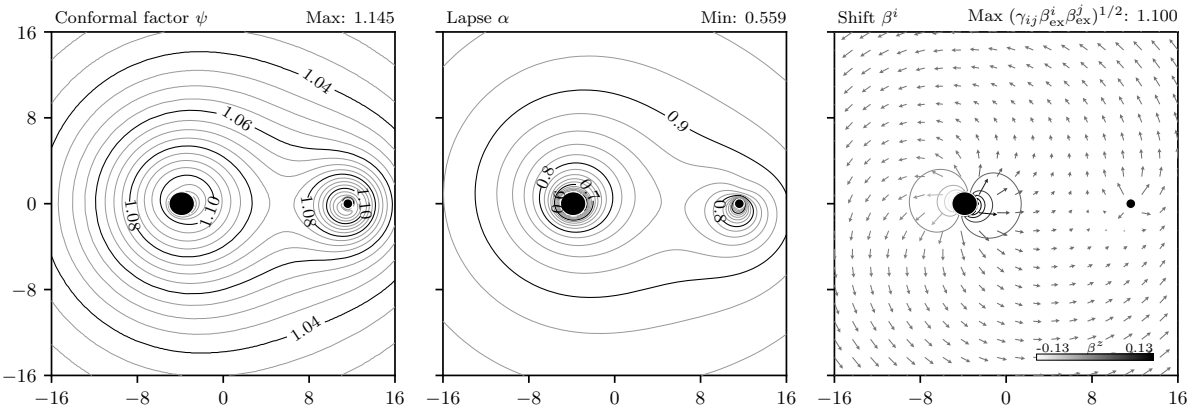


Figure 4.19: Spinning and unequal-mass SKS initial data with negative-expansion boundary conditions. The spinning black hole (left) is represented by a nonspherical excision boundary. Since its spin χ_1 is tilted toward positive y, the excision boundary is slightly elongated horizontally in this plot.

by transitioning to a spherical shell in the vicinity of the left black hole (see discussion in Section 4.1.2). Figure 4.18 illustrates these effects. I have not reanalyzed the domain based on power monitors.

Figure 4.19 shows a visualization of a two-dimensional slice through the initial data. Note that the addition of a spin unaligned with the orbital axis has broken the symmetry with respect to the orbital plane. This is evident in the right panel of Fig. 4.19, where the shift component β^z is nonzero around the spinning (left) black hole. Therefore, a fully three-dimensional formalism is required to solve this configuration.

Figure 4.20 (left panel) presents the convergence of the solution under hp refinement of the domain. The circled configuration corresponds to the domain depicted in Fig. 4.18. I compute the discretization error as detailed in Section 2.4.4, by interpolating the solution variables to a set of sample points $\boldsymbol{x}_m$ and computing the L_2 norm of the difference to a high-resolution reference run, Eq. (2.59). To account for the loss of symmetry I choose sample points near both excisions, specifically $\boldsymbol{x}_1 = (-5.14, 0, 0)$ (near the left excision), $\boldsymbol{x}_2 = (12.11, 0, 0)$ (near the right excision), $\boldsymbol{x}_3 = (0, 0, 0)$ (origin), and $\boldsymbol{x}_3 = (100, 0, 0)$ (far field). The discretization error $\|u - u_{\rm ref}\|$ converges exponentially, and at a slightly lower rate than for the equal-mass and nonspinning configuration (Fig. 4.16). Also shown is the exponential convergence of the constraint norm discussed in Section 4.2.4 below.

The right panel of Fig. 4.20 shows the convergence of quantities extracted on the apparent horizons of the two black holes. I find the apparent horizons numerically and measure their quantities using the methods laid out in Section 4.1.2. They converge exponentially for both black holes, following the convergence of the discretization error plotted in the left panel. I measure a Christodoulou mass of $M_1 \approx 0.987$ and a dimensionless spin magnitude of $\chi_1 \approx 0.925$ for the left black hole, and $M_2 \approx 0.321$ and $\chi_2 \approx 0.013$ for the right black hole. These values deviate from the parameters imposed on the background Kerr-Schild solutions, as discussed in Section 4.1.2.

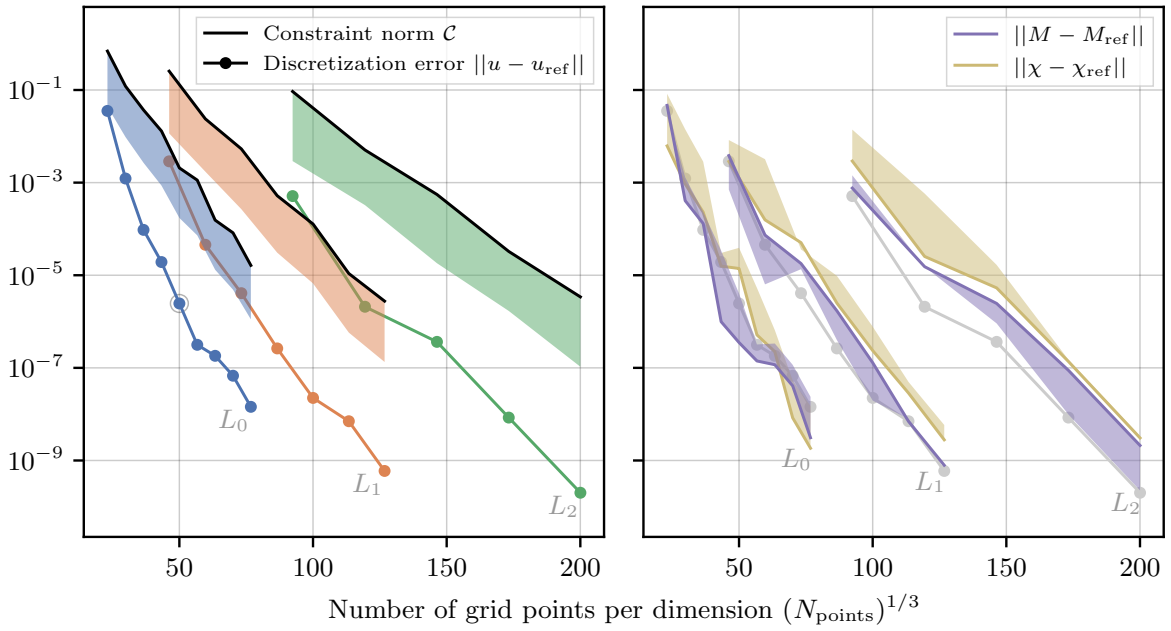


Figure 4.20: *Left:* Convergence of the spinning SKS initial data, Fig. 4.19, at three h-refinement levels $L \in [0, 2]$. The constraint norm $\mathcal{C}$ and the shaded area below are defined in Section 4.2.4. *Right:* Convergence of apparent horizon quantities. The light gray lines illustrate the discretization error, as shown in the left panel.

4.2.4 Constraint norms

The Hamiltonian and momentum constraints, Eqs. (1.42a) and (1.42b), can be evaluated numerically to indicate how accurately an initial data slice satisfies the Einstein equations. Since we solve them numerically as part of the XCTS equations, Eq. (1.56), we expect their residual to converge to zero with increasing numerical resolution. I define the Hamiltonian and momentum constraints as

$$\mathcal{C}_{\text{H}} = \frac{1}{2}\left(R + K^2 - K_{ij}K^{ij}\right) - 8\pi\rho_{\text{H}}, \tag{4.22a}$$

$$\mathcal{C}^i_{\text{M}} = \nabla_j\left(K^{ij} - \gamma^{ij}K\right) - 8\pi S^i, \tag{4.22b}$$

including a factor of $1/2$ in the Hamiltonian constraint for consistency with `SpEC` and to scale the source terms in both constraints the same. I also define the combined constraint norm

$$\mathcal{C} = \sqrt{\mathcal{C}^2_{\text{H}} + \sum_{i=1}^{3} |\mathcal{C}^i_{\text{M}}|^2}. \tag{4.23}$$

The constraint norm is plotted alongside the discretization error for the spinning single black hole problem in Fig. 4.7, and for the spinning BBH problem in Fig. 4.20. In both cases, the solid black line represents the L_2 norm over all grid points in the domain. As expected, the constraint norm converges exponentially, and with a lower rate than the discretization error because it involves two numerical derivatives.[24] Note that the constraints are not normalized to a dimensionless quantity, as is done, for instance, by Varma, Scheel, and Pfeiffer [78].

While the constraints converge exponentially as expected, their L_2 norm

24: See Fig. 6.3 in Ref. [170] for a similar result.

[170]: Pfeiffer (2003), *Initial Data for Black Hole Evolutions*

[78]: Varma, Scheel, and Pfeiffer (2018), *Comparison of binary black hole initial data sets*

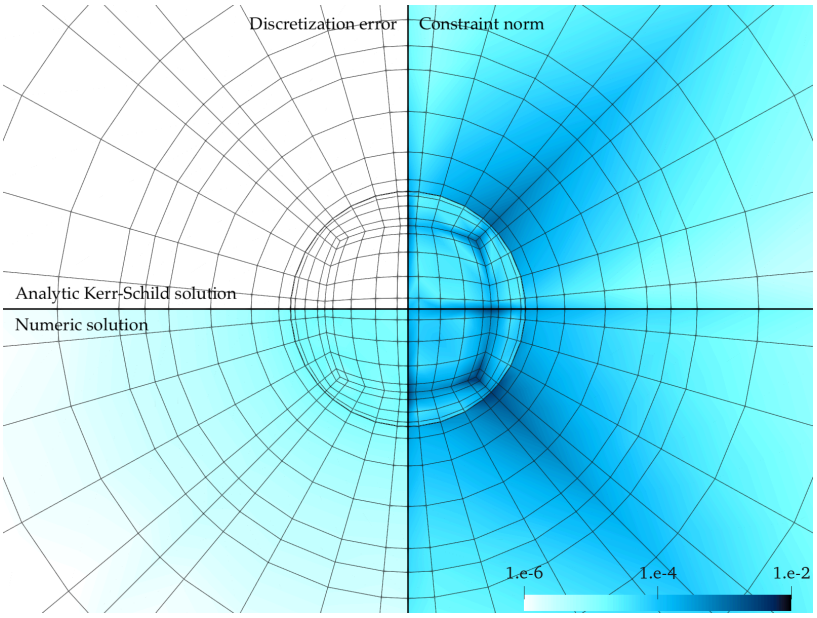


Figure 4.21: Cut through nonspinning Kerr-Schild data on a spherical shell domain. *Upper half:* Analytic solution evaluated on the grid. *Lower half:* Numeric DG-discretized solution. *Left half:* Discretization error $|u - u_{\text{analytic}}|$, Eq. (4.24). The upper-left panel is identically zero. *Right half:* Constraint norm $\mathcal{C}$, Eq. (4.23). The constraint norm accumulates error at element boundaries on the cubed-sphere grid, with little difference between the analytic and numeric solution.

over grid points is larger when computed on the cubed-sphere grid in `SpECTRE` than on the spherical shells in `SpEC`. This holds for data with the same discretization error on both grids. For instance, the three highest-resolution configurations in Fig. 4.16 agree with each other to $\lesssim 10^{-7}$ at the sample points. However, the L_2 constraint norm over all grid points in `SpEC` is $\sim 10^{-6}$ while it is $\sim 10^{-3}$ in `SpECTRE`, three magnitudes larger.

The reason for the large L_2 constraint norms on our cubed-sphere grid is that numerical errors from the two numerical derivatives accumulate toward element boundaries, and get amplified by the nonlinearity in the Hamiltonian and momentum constraints, Eq. (4.22). Figure 4.21 illustrates this effect. It shows a logarithmically scaled close-up of the discretization error and the constraint norm in the vicinity of an analytic Kerr-Schild solution. The discretization error is computed to the analytic solution as

$$|u - u_{\text{analytic}}| = \sqrt{\sum_A \left(u_A - u_{A,\text{analytic}}\right)^2}, \tag{4.24}$$

where $u_A = \{\psi, \alpha\psi, \beta^i\}$ for this XCTS problem. The upper left panel of Fig. 4.21 confirms that the discretization error is identically zero when the analytic solution is evaluated on the computational grid. The lower left panel shows the discretization error of the numerical solution, with choices identical to those made in Section 2.4.3.[25] It provides a measure of the accuracy with which the computational domain can represent the solution, and is $\lesssim 10^{-5}$ throughout the grid. This is consistent with the circled configuration in Fig. 2.11. The right panels show the constraint norm $\mathcal{C}$, Eq. (4.23). It accumulates errors up to three orders of magnitude larger than the discretization error at the element boundaries, and particularly at the element corners. Note that the cutting plane in the illustration is also an element boundary (in the third dimension). The constraint norm computed from the analytic data (top) and the numeric solution (bottom) show no significant difference.

The L_2 constraint norm over grid points emphasizes the error at element boundaries further, since LGL grid point spacing reduces quadratically toward the edges. A volume integral norm in element-logical coordinates

25: Input file `KerrSchildConstraints.yaml` (Appendix B.7)

can cure this over-emphasis because it weights grid points with their quadrature weight, but the errors at element boundaries and corners will still dominate the norm. To quantify this effect I also plot the L_2 constraint norm over only the interior grid points of all elements as colored bands in Figs. 4.7 and 4.20, excluding grid points on element boundaries. It converges exponentially with the same rate as the L_2 constraint norm over all grid points, but remains about one order of magnitude smaller.

Evidently, the L_2 constraint norm over grid points converges exponentially, but is not a reliable measure of the discretization error on our cubed-sphere LGL grids. Instead, both the convergence of field values at sample points and the convergence of apparent horizon quantities provide measures of the discretization error that are consistent with each other and with `SpEC`. In the future we may explore shells with spherical harmonic basis functions in `SpECTRE`, which potentially resolve this effect by eliminating element boundaries in angular directions.

4.3 Single TOV stars

In this section I solve for isolated Tolman-Oppenheimer-Volkoff (TOV) stars with the XCTS formalism, exploring the applicability of the discontinuous Galerkin scheme and my new elliptic solver to problems involving neutron stars. I employ a polytropic equation of state in this section, but none of the studies is specific to this choice. The polytropic equation of state is [26]

$$P(\rho_0) = K\rho_0^\Gamma, \tag{4.25}$$

where K is the polytropic constant, and Γ is the polytropic exponent related to the polytropic index n by $\Gamma = 1 + 1/n$. Ongoing work is concerned with implementing a range of analytic and tabulated equations of state in `SpECTRE`.

26: Eq. (1.86) in Baumgarte and Shapiro [17], or Eq. (2.242) in Rezzolla and Zanotti [114].

4.3.1 Integration schemes for the TOV equations

Before solving TOV stars with the XCTS formalism in three dimensions I compute TOV profiles using conventional one-dimensional integration schemes for comparison. To this end, I develop an extension of the TOV integration scheme by Lindblom [225] to isotropic coordinates in this section.

[225]: Lindblom (1998), *Phase transitions and the mass-radius curves of relativistic stars*

The TOV equations are a set of coupled ordinary differential equations (ODEs) representing a spherically symmetric perfect fluid in hydrostatic equilibrium.[27] The spherically symmetric and static line element is [28]

$$\mathrm{d}s^2 = -e^{2\Phi_t}\,\mathrm{d}t^2 + e^{2\Phi_r}\,\mathrm{d}r^2 + e^{2\Phi_\Omega} r^2\,\mathrm{d}\Omega^2\,, \tag{4.26}$$

parametrized by the metric potentials Φ_t, Φ_r, and Φ_Ω. The 3+1 quantities

27: See, e.g., Sec. 1.3 in Baumgarte and Shapiro [17] or Sec. 12.1 in Rezzolla and Zanotti [114].

28: See Sec. 1.3 and Chapter 8 in Baumgarte and Shapiro [17].

in this parametrization are

$$\alpha = e^{\Phi_t}, \tag{4.27a}$$
$$\beta^i = 0, \tag{4.27b}$$
$$\gamma_{ij} = e^{2\Phi_r}\delta_{ij} + \left(e^{2\Phi_r} - e^{2\Phi_\Omega}\right)\frac{x^i x^j}{r^2} \tag{4.27c}$$

in Cartesian coordinates.

Schwarzschild (areal) coordinates

The canonical TOV solution employs *areal* coordinates where we choose

$$e^{2\Phi_\Omega} = 1, \tag{4.28}$$

so the solution outside the star is Schwarzschild spacetime in standard Schwarzschild coordinates. We also reparametrize the radial metric potential as [29]

$$e^{2\Phi_r} = \left(1 - \frac{2m(r)}{r}\right)^{-1}, \tag{4.29}$$

defining the *interior mass* $m(r)$. Then, the Einstein and Euler equations with a perfect fluid source reduce to the standard TOV equations [30]

$$\frac{\mathrm{d}m}{\mathrm{d}r} = 4\pi r^2 \rho, \tag{4.30a}$$
$$\frac{\mathrm{d}P}{\mathrm{d}r} = -(\rho + P)\frac{m/r^2 + 4\pi r P}{1 - 2m/r}, \tag{4.30b}$$
$$\frac{\mathrm{d}\Phi_t}{\mathrm{d}r} = -(\rho + P)^{-1}\frac{\mathrm{d}P}{\mathrm{d}r}, \tag{4.30c}$$

where ρ is the energy density and P is the pressure. The fluid variables are closed by an (isentropic) equation of state $P(\rho_0)$.

To integrate the TOV equations in areal coordinates we follow the strategy laid out by Lindblom [225]. We define the variables

$$u \coloneqq r^2 \quad \text{and} \quad v \coloneqq \frac{m(r)}{r} \tag{4.31}$$

to rewrite the TOV equations as [31]

$$\frac{\mathrm{d}u}{\mathrm{d}\ln h} = -\frac{2u(1-2v)}{4\pi u P + v} \tag{4.32a}$$
$$\frac{\mathrm{d}v}{\mathrm{d}\ln h} = -(1-2v)\frac{4\pi u\rho - v}{4\pi u P + v}. \tag{4.32b}$$

At the center of the star, $u = 0 = v$, Eq. (4.32) has the limits [32]

$$\left.\frac{\mathrm{d}u}{\mathrm{d}\ln h}\right|_{r=0} = -\frac{3}{2\pi(\rho_c + 3P_c)}, \tag{4.33a}$$
$$\left.\frac{\mathrm{d}v}{\mathrm{d}\ln h}\right|_{r=0} = -\frac{2\rho_c}{\rho_c + 3P_c}. \tag{4.33b}$$

Here, ρ_c is the central density of the star chosen to parametrize all solutions. It determines all other fluid variables at the center of the star,

29: See Eq. (1.76) and Eq. (1.80) in Baumgarte and Shapiro [17]. Note that Eq. (1.76) is missing a factor of 2 in the exponent.

30: Eqs. (1.77) to (1.79) in Baumgarte and Shapiro [17].

[225]: Lindblom (1998), *Phase transitions and the mass-radius curves of relativistic stars*

31: See Eqs. (A2) and (A3) in Lindblom [225], and note that the quantity h in Ref. [225] is actually the logarithm of the specific enthalpy, which we denote $\ln h$ here.

32: Eqs. (A7) and (A8) in Lindblom [225]

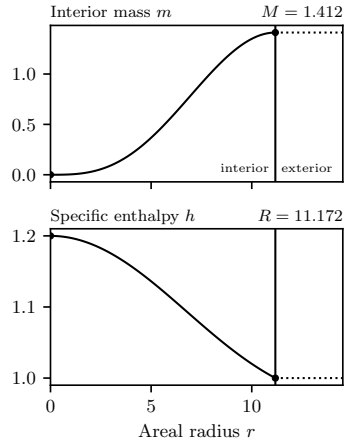


Figure 4.22: TOV profile for the central enthalpy $h_c = 1.2$, and a polytropic equation of state with constant $K = 123.6489$ and exponent $\Gamma = 2$.

such as the central pressure P_c and the central enthalpy h_c, using the equation of state.

We integrate the TOV equations, Eq. (4.32), from the center of the star to its surface. The choice of variables in Eq. (4.32) allows us to integrate numerically in the fixed interval $\ln h \in [\ln h_c, 0]$, evaluating the areal radius r and $m(r)/r$ along the way. Therefore, the result of the integration is the radial profile $\ln h(r)$ and $m(r)/r$ in the interior of the star, as well as the areal radius R of its surface, as illustrated in Fig. 4.22. We can obtain all fluid variables from the enthalpy profile and the equation of state, the radial metric potential from $m(r)/r$, and the temporal metric potential $\Phi_t = \ln \alpha$ from Eq. (4.30c),[33]

$$\Phi_t - \Phi_{t,R} = -\int_0^P \frac{\mathrm{d}P'}{\rho + P'} = -\ln h. \tag{4.34}$$

Here, $\Phi_{t,R}$ is the value of $\Phi_t(r)$ at $r = R$ that we now match to the exterior solution.

Outside the star the spacetime is Schwarzschild with mass parameter

$$M = m(R) = \int_0^R 4\pi r^2 \rho \, \mathrm{d}r \,. \tag{4.35}$$

From Eq. (4.30c) we recover the Schwarzschild lapse outside the star,

$$\Phi_t(r) = \frac{1}{2}\ln(1 - 2M/r), \quad r \geq R, \tag{4.36}$$

where we have chosen the integration constant so $\Phi_t = 0$ as $r \to \infty$. Therefore, the matching constant in Eq. (4.34) is

$$\Phi_{t,R} = \frac{1}{2}\ln(1 - 2M/R). \tag{4.37}$$

Note that the matching constant is the injection energy [34]

$$\mathcal{E} = -hk^\mu u_\mu = h\alpha = \alpha_R = e^{\Phi_{t,R}}, \quad r \leq R. \tag{4.38}$$

It is the Bernoulli conserved quantity along streamlines $u^\mu = u^t t^\mu$ associated with the Killing vector $k^\mu = t^\mu$ of this static problem. The second equality in Eq. (4.38) follows from the normalization $u^\mu u_\mu = -1$, so $u^t = 1/\alpha$ and $u_t = -\alpha$. The third equality holds because the injection energy is conserved within the static star, $\boldsymbol{\nabla}_\mu \mathcal{E} = 0$, since streamlines are aligned with the Killing vector.[35] This approach based on conservation laws also reproduces Eq. (4.34).

33: Eq. (A1) in Lindblom [225]

34: Eq. (3.99) in Rezzolla and Zanotti [114]

35: See Eq. (4.5) in Moldenhauer et al. [87].

[87]: Moldenhauer et al. (2014), *Initial data for binary neutron stars with adjustable eccentricity*

Transformation to isotropic coordinates

Since the TOV solution is spherically symmetric, we can transform it to isotropic coordinates $\bar{x}$ in which the spatial metric is conformally flat. This strategy is employed in many numerical codes, but rarely detailed in the literature.[36] Here, I describe the TOV integration scheme that I have developed and implemented in `SpECTRE` to extend the scheme by Lindblom [225] to isotropic coordinates.

36: The Einstein toolkit includes some notes by Baumgarte [226]. Tsokaros, Uryū, and Rezzolla [110] describe a similar procedure in Appendix B. Clark and Laguna [100] detail a procedure based on a symmetry reduction of the XCTS equations.

[100]: Clark and Laguna (2016), *Bowen-York Type Initial Data for Binaries with Neutron Stars*

[110]: Tsokaros, Uryū, and Rezzolla (2015), *New code for quasiequilibrium initial data of binary neutron stars: Corotating, irrotational, and slowly spinning systems*

[226]: Baumgarte (2009), *Oppenheimer-Volkov solution in isotropic coordinates*

We reparametrize the spatial metric potentials in terms of the conformal factor ψ,

$$e^{2\Phi_{\bar{\Omega}}} = e^{2\Phi_{\bar{r}}} = \psi^4, \tag{4.39}$$

so the spatial metric, Eq. (4.27c), reduces to the conformally flat form

$$\gamma_{ij} = \psi^4 \delta_{ij} \tag{4.40}$$

in Cartesian coordinates. We only transform coordinates radially, so $\mathrm{d}\bar{\Omega} = \mathrm{d}\Omega$ and $\mathrm{d}\bar{t} = \mathrm{d}t$. Comparing the line element, Eq. (4.26), in areal and isotropic coordinates we find

$$\psi^2 = \frac{r}{\bar{r}} \tag{4.41}$$

from the angular component and

$$\frac{\mathrm{d}\bar{r}}{\bar{r}} = \left(1 - 2m(r)/r\right)^{-1/2} \frac{\mathrm{d}r}{r} \tag{4.42}$$

from the radial component. Therefore, the conformal factor parametrizes the coordinate transformation between areal and isotropic radius, and is determined by the ODE (4.42).

Outside the star, the mass is $m(r) = M$ for $r \geq R$. Therefore, we can integrate Eq. (4.42) analytically to obtain the standard Schwarzschild solution in isotropic coordinates,[37]

37: Eq. (31.23) in Misner, Thorne, and Wheeler [227]

[227]: Misner, Thorne, and Wheeler (1973), *Gravitation*

$$r = \bar{r}\left(1 + \frac{M}{2\bar{r}}\right)^2, \tag{4.43a}$$

$$\text{or} \quad \bar{r} = \frac{1}{2}\left(\sqrt{r^2 - 2Mr} + r - M\right), \tag{4.43b}$$

$$\text{or} \quad \psi = 1 + \frac{M}{2\bar{r}}, \tag{4.43c}$$

where we fixed the integration constant so areal and isotropic radius coincide as $r \to \infty$.

In order to integrate Eq. (4.42) inside the star, we rewrite the equation as an ODE for the conformal factor ψ following Baumgarte [226],

[226]: Baumgarte (2009), *Oppenheimer-Volkov solution in isotropic coordinates*

$$\frac{\mathrm{d}\bar{r}}{\bar{r}} - \frac{\mathrm{d}r}{r} = \mathrm{d}\ln\bar{r} - \mathrm{d}\ln r = -2\,\mathrm{d}\ln\psi \tag{4.44a}$$

$$= \frac{1 - \sqrt{1 - 2m(r)/r}}{\sqrt{1 - 2m(r)/r}} \frac{\mathrm{d}r}{r}. \tag{4.44b}$$

At this point we deviate from the literature. We employ the variables $u = r^2$ and $v = m(r)/r$, Eq. (4.31), as well as $\mathrm{d}\ln h = -\mathrm{d}\Phi_t$, Eq. (4.34), to rewrite the ODE for the conformal factor as

$$\frac{\mathrm{d}\ln\psi}{\mathrm{d}\ln h} = \frac{\sqrt{1 - 2v}}{1 + \sqrt{1 - 2v}} \frac{v}{4\pi u P + v}, \tag{4.45}$$

using the TOV equation for the pressure, Eq. (4.30b). At the stellar surface, Eq. (4.45) has the limit

$$\left.\frac{\mathrm{d}\ln\psi}{\mathrm{d}\ln h}\right|_{r=0} = \frac{1}{2}\frac{\rho_c}{\rho_c + 3P_c}. \tag{4.46}$$

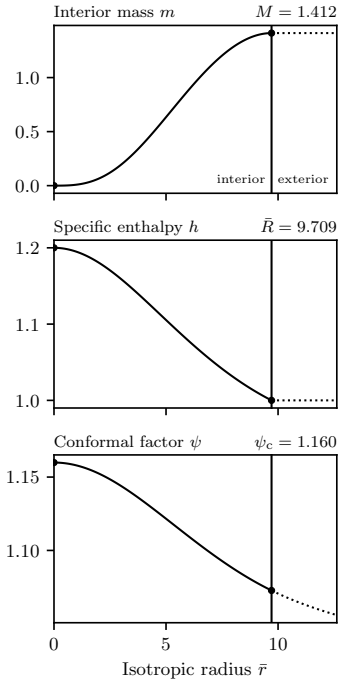


Figure 4.23: TOV profile in isotropic coordinates for the same parameters listed in Fig. 4.22.

It can be integrated numerically from the center of the star to the surface alongside the Lindblom TOV equations, Eqs. (4.32) and (4.33). The TOV equations are independent of the conformal factor, but the side-by-side integration eliminates the need for a second numeric integration. It also allows the use of the variables u and v in Eq. (4.45) directly at the (adaptive) integration steps, instead of interpolating them afterwards.

Because Eq. (4.45) is an ODE for the logarithm of ψ, it leaves an overall multiplicative factor undetermined. We begin the numeric integration with $\ln \psi = 0$ at the center of the star. After the integration is complete, we rescale

$$\psi(r) \;\to\; \psi(r)\,\frac{1 + M/(2\bar{R})}{\psi(R)}, \tag{4.47}$$

to achieve continuity at the stellar surface with the exterior conformal factor, Eq. (4.43c). Here, the outer isotropic radius of the star, $\bar{R}$, is determined from the outer areal radius R and the total mass M by Eq. (4.43b). Fig. 4.23 shows a TOV profile in isotropic coordinates obtained from this integration scheme.

Fluid variables are determined from the enthalpy profile and the equation of state, just like they are in areal coordinates. The temporal metric potential Φ_t is also still given by Eq. (4.34) in the interior of the star, and by Eq. (4.36) in the exterior. Spatial metric potentials are determined by the conformal factor, Eq. (4.39). Where needed, the conformal factor can also be used to transform between areal and isotropic radius, Eq. (4.41).

4.3.2 TOV star solutions to the XCTS equations

In preparation for initial data involving neutron stars I study single TOV stars using the full three-dimensional XCTS formalism with matter. Of particular interest is the applicability of the discontinuous Galerkin (DG) scheme to the TOV problem, since the solution is not smooth at the stellar surface.

To explore the effect of the stellar surface, I source the XCTS equations with a static TOV matter profile obtained from the ODE integration scheme detailed in Section 4.3.1 above. I impose conformal flatness,

$$\bar{\gamma}_{ij} = \eta_{ij}, \tag{4.48}$$

and maximal slicing,

$$K = 0, \tag{4.49}$$

consistent with the TOV solution in isotropic coordinates, as well as the matter sources (1.43),

$$\rho_{\mathrm{H}} = \rho, \quad S^i = 0, \quad \text{and} \quad S = 3P. \tag{4.50}$$

I find it unnecessary to conformally scale the matter sources for this problem.[38] I solve the XCTS equations on a spherical domain and impose the TOV solution for ψ, $\alpha\psi$, and β^i as Dirichlet conditions at the outer boundary.

Figure 4.24 shows the convergence of the DG discretization error to the TOV solution with increasing resolution.[39] Since the matter profile is not smooth at the surface of the star (see Figs. 4.22 and 4.23), the

38: Baumgarte, Ó Murchadha, and Pfeiffer [228] suggested a conformal matter scale of eight, so $\bar{\rho}_{\mathrm{H}} = \rho_{\mathrm{H}}^8$. Many recent studies use a scale of six instead, such as Refs. [108, 112]. Papenfort et al. [89] use no conformal matter scale at all.

[89]: Papenfort et al. (2021), *New public code for initial data of unequal-mass, spinning compact-object binaries*
[108]: Rashti et al. (2021), *Elliptica: a new pseudo-spectral code for the construction of initial data*
[112]: Tacik et al. (2016), *Initial data for black hole–neutron star binaries, with rotating stars*
[228]: Baumgarte, Ó Murchadha, and Pfeiffer (2007), *The Einstein constraints: Uniqueness and non-uniqueness in the conformal thin sandwich approach*

39: Input file `Tov.yaml` (Appendix B.8)

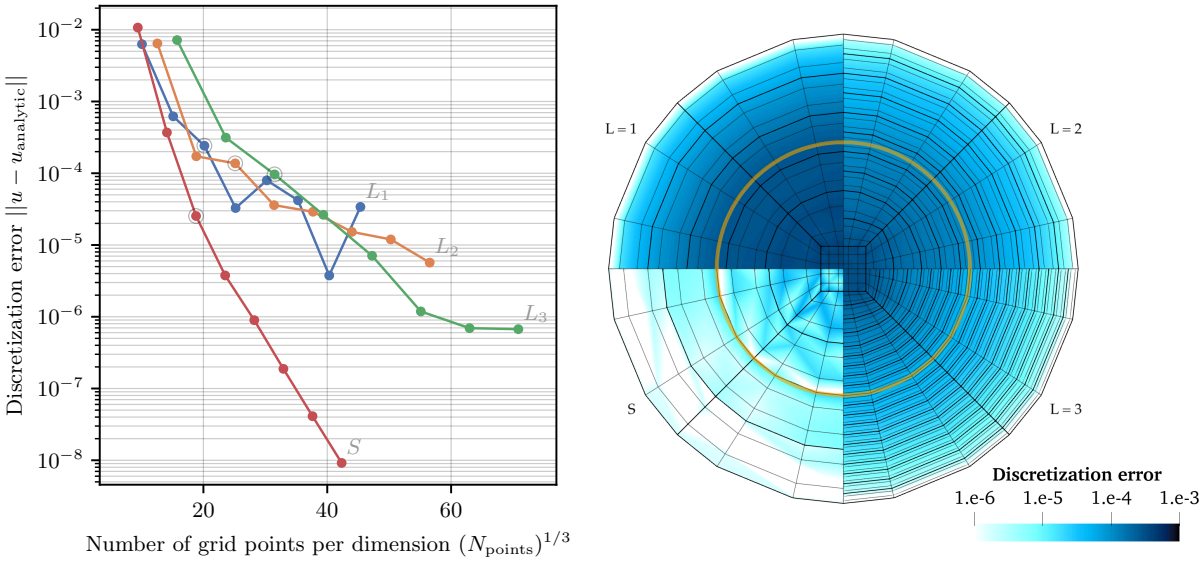


Figure 4.24: Convergence of the TOV problem with the DG-discretized XCTS formalism in three dimensions. *Left:* The discretization error converges polynomially with radial h refinement when the stellar surface lies within a DG element ($L = 1$ to $L = 3$). Exponential convergence is recovered when the stellar surface coincides with a block boundary (S). All four curves cover polynomial degrees $P \in [1, 8]$ per element and dimension (see input file `Tov.yaml`). *Right:* Domain and discretization error for the four circled configurations in the left panel. The configurations $L = 1$ to $L = 3$ transition from a cube covering the origin to a sphere at $r = 6$ (first black inner ring in upper left quarter). The orange ring indicates the position of the stellar surface at $R = 9.709$, where the configuration S transitions to a sphere.

DG discretization scheme (or any spectral scheme) loses exponential convergence when the surface lies within an element. However, we can still radially h refine the domain to reduce the error polynomially. An adaptive mesh refinement (AMR) algorithm, such as developed in Ref. [8], may take advantage of the smoothness of the solution in all elements but those covering the stellar surface, and only h refine the latter.

[8]: Vincent, Pfeiffer, and **Fischer** (2019), *hp-adaptive discontinuous Galerkin solver for elliptic equations in numerical relativity*

Figure 4.24 also shows that the DG scheme recovers exponential convergence when the stellar surfaces is placed at a grid boundary. This is straightforward for a TOV problem where the position of the stellar surface is known, but nontrivial in initial data problems with orbiting, spinning, and tidally deformed stars. Many contemporary spectral codes employ *surface-fitted coordinates* to deform grid boundaries such that they conform to the numerically determined stellar surfaces, and update the grid deformation accordingly throughout the elliptic solver procedure [107, 182]. A subject of future work will be to develop a surface-fitting algorithm for our DG scheme, an AMR algorithm to resolve the surface using h refinement, or both. The technology to deform excision surfaces to black hole horizons detailed in Section 4.1.2 can also be used to conform grid boundaries to stellar surfaces. In Chapter 5 I present encouraging results applying our DG scheme to another discontinuous problem.

[107]: Dietrich et al. (2015), *Binary Neutron Stars with Generic Spin, Eccentricity, Mass ratio, and Compactness - Quasi-equilibrium Sequences and First Evolutions*
[182]: Foucart et al. (2008), *Initial data for black hole-neutron star binaries: A Flexible, high-accuracy spectral method*

4.4 Preview: binary neutron stars

We can solve for binary neutron star (BNS) initial data with a procedure similar to binary black holes, meaning that we impose background quantities by a superposition of isolated solutions and solve for the XCTS

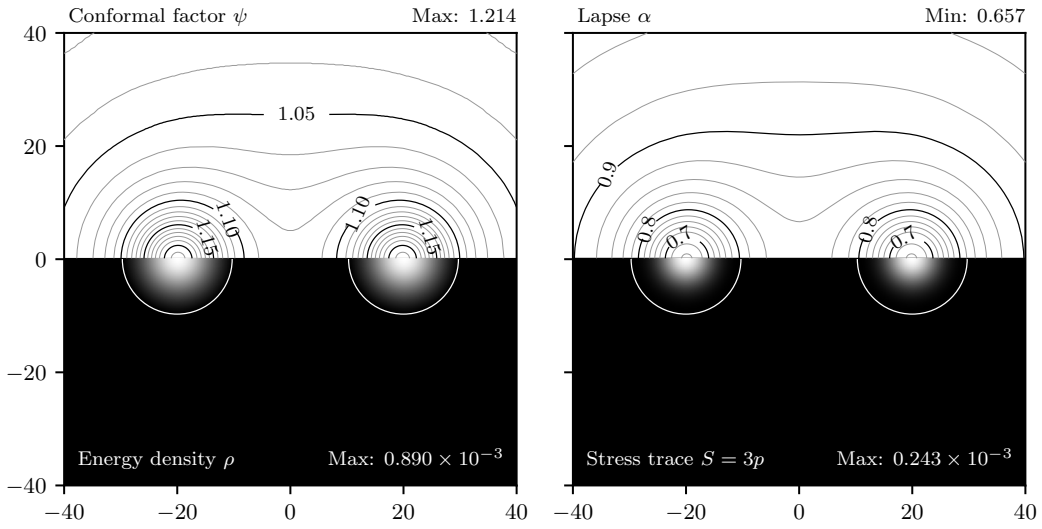


Figure 4.25: Head-on BNS initial data with two TOV stars that are initially at rest. The density and pressure profiles (bottom) determine the gravity sector (top). Note that $\beta^i = 0$ for this problem.

variables ψ, $\alpha\psi$, and β^i in three dimensions with no symmetry assumptions. At the time of writing, I have implemented the following isolated matter solutions in `SpECTRE` for composing conformal backgrounds:

TOV The TOV solution in standard Schwarzschild (areal) coordinates (see Section 4.3.1). It is maximally sliced, $K = 0$.

Isotropic TOV The TOV solution in isotropic coordinates, which is both conformally flat, $\gamma_{ij} = \eta_{ij}$, and maximally sliced, $K = 0$ (see Section 4.3.1).

Constant-density star A spherical star with constant density, assuming a moment of time symmetry, $K_{ij} = 0$, and conformal flatness, $\gamma_{ij} = \eta_{ij}$. This solution is described in detail by Baumgarte, Ó Murchadha, and Pfeiffer [228], and also in Exercise 3.8 in Baumgarte and Shapiro [17], since it exhibits nonuniqueness properties that are typical for the XCTS system.

[228]: Baumgarte, Ó Murchadha, and Pfeiffer (2007), *The Einstein constraints: Uniqueness and non-uniqueness in the conformal thin sandwich approach*

[17]: Baumgarte and Shapiro (2010), *Numerical Relativity: Solving Einsteins Equations on the Computer*

These solutions can be superposed with any other vacuum or nonvacuum solution to compose the conformal metric and the trace of the extrinsic curvature by Eq. (2.55).

With neutron stars we have no excision surfaces that require boundary conditions, but we need to handle the fluid sources and their dynamics as described in Section 1.2.4. A particularly simple case is initial data representing two TOV stars initially at rest, some coordinate distance D_0 apart, which will approach each other head-on and collide when evolved. We can solve for the gravity sector of this scenario with the XCTS equations when we source them with the density and pressure profile of two isolated TOV stars, Eq. (4.50).

Figure 4.25 shows a solution for the gravity induced by the density and pressure profile of two isotropic TOV stars at rest.[40] The conformal factor and lapse are axisymmetric (though the problem was solved in three dimensions) and show the gravitational interaction between the two bodies. The shift is zero, $\beta^i = 0$, since this spacetime is maximally sliced and $S^j = 0$, so the momentum constraint (1.56c) is trivially satisfied.

40: Input file `BnsHeadOn.yaml` (Appendix B.9)

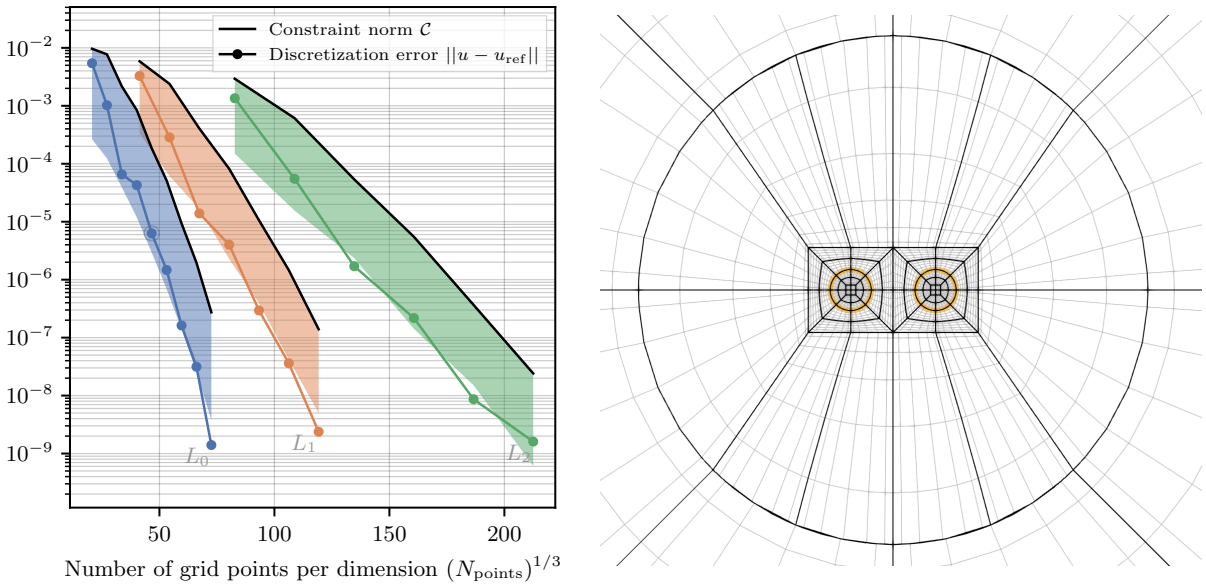


Figure 4.26: Convergence (left) and computational domain (right) for the head-on BNS problem. The constraint norm $\mathcal{C}$ and the shaded area below are defined in Section 4.2.4. Note that $\beta^i = 0$ so $\mathcal{C} = \mathcal{C}_{\text{H}}$. The discretization error is measured by Eq. (2.59) with the sample points $\boldsymbol{x}_1 = (-20, 0, 0)$ (star center) and $\boldsymbol{x}_2 = (0, 0, 0)$ (origin). The orange rings in the right panel indicate the position of the stellar surfaces. The outer boundary is at $r = 600$ and imposes flatness ($\psi = 1$ and $\alpha = 1$).

Figure 4.26 shows the convergence of the discretization error and the domain configuration. We recover exponential convergence by placing grid boundaries at the neutron star surfaces, as detailed in Section 4.3.

Such superposed and constraint-solved initial data represent valid solutions to the Einstein and Euler equations, but will only approximate astrophysical scenarios. Due to tidal interactions between the binary components, the superposed matter profiles will not be in hydrostatic equilibrium initially. We can use these solutions to seed experimental head-on BNS evolutions with the GRMHD part of `SpECTRE`, and simultaneously work on more realistic astrophysical initial data that treat the matter sector dynamically (see Section 1.2.4).

Figure 4.27 presents the convergence of the Newton-Krylov elliptic solver for the head-on BNS problem. Starting at an initial guess constructed by superposing the two isolated TOV solutions, the Newton-Raphson algorithm converges in five steps, each with up to eleven multigrid-Schwarz preconditioned GMRES iterations. This configuration completed in 32 seconds on 60 cores. The highest-resolution $L = 0$ configuration pictured in Fig. 4.26, which reaches a discretization error of $\lesssim 10^{-8}$, completed in ≈ 3.5 minutes on 60 cores. Even the $L = 2$ configuration with comparable discretization error and over 200 grid points per dimension completed in under ten minutes on 240 cores, and also needed no more than five Newton-Raphson steps with up to eleven GMRES iterations each. This scale independence of the convergence behavior is a feature of the multigrid-Schwarz preconditioning (see discussion in Section 3.4.1 and Fig. 3.12). It is particularly promising for problems involving equations of state with phase transitions, where h refinement is advantageous.

Looking ahead, we will solve for the equilibrium matter profile alongside the gravity sector as discussed in Section 1.2.4. Contemporary studies of single rotating stars, BNS, and BHNS initial data typically employ an

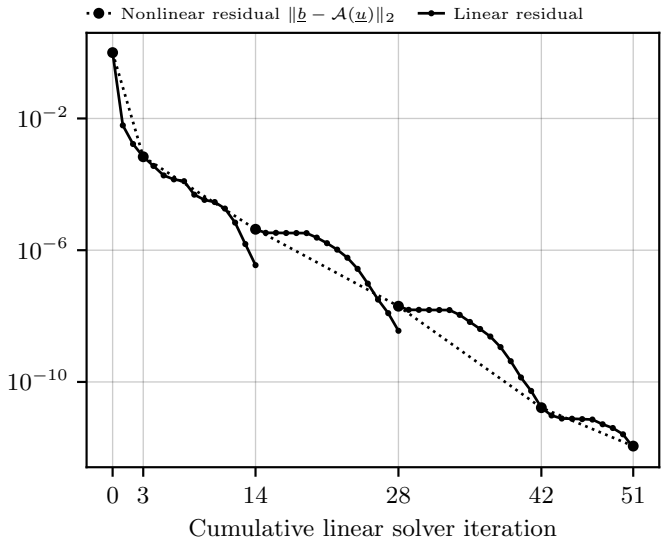


Figure 4.27: Convergence of the elliptic solver for the head-on BNS configuration circled in Fig. 4.26.

iteration procedure of varying complexity.[41] They solve the Einstein and Euler equations along with updates to the grid deformation, control of intrinsic parameters such as masses and spins of the object, and control of extrinsic parameters such as eccentricity, orbital angular velocity, linear momentum, and center of mass. These iteration procedures typically invoke the elliptic solver tens to hundreds of times, and can take hours to days to complete. Therefore, accelerated elliptic solves parallelized to computing clusters have the potential to speed up such iteration procedures considerably.

Furthermore, the iterations are often damped, advancing variables only by step lengths of 10 % to 50 % [87, 108, 110, 112, 183, 230]. Therefore, advancements to the iteration procedure have considerable potential to accelerate initial data generation. For example, I am confident that the elliptic solver technology I present in this thesis is capable of solving the velocity potential equation, Eq. (1.63), alongside the XCTS equations instead of solving the equations successively. A possible subject of future work is to explore iteration procedures that leverage this capability to provide more accurate updates, and hence the potential for reduced damping. More accurate iterations also have the potential to increase the robustness of these procedures, which can be prone to failure, e.g., for highly-compact neutron stars [231].

41: Tacik et al. [112] solve the XCTS equations together, and the velocity potential equation separately. References [81, 108, 183] solve all equations separately and in succession. See also Price, Markakis, and Friedman [229] for a convergence analysis of an iteration procedure for Newtonian stars. Tsokaros, Uryū, and Rezzolla [110] illustrate the iteration procedure of the `COCAL` code in Appendix C.

[81]: Uryū et al. (2016), *New code for equilibriums and quasiequilibrium initial data of compact objects. III. Axisymmetric and triaxial rotating stars*
[87]: Moldenhauer et al. (2014), *Initial data for binary neutron stars with adjustable eccentricity*
[108]: Rashti et al. (2021), *Elliptica: a new pseudo-spectral code for the construction of initial data*
[110]: Tsokaros, Uryū, and Rezzolla (2015), *New code for quasiequilibrium initial data of binary neutron stars: Corotating, irrotational, and slowly spinning systems*
[112]: Tacik et al. (2016), *Initial data for black hole–neutron star binaries, with rotating stars*
[183]: Tichy et al. (2019), *Constructing binary neutron star initial data with high spins, high compactnesses, and high mass ratios*
[229]: Price, Markakis, and Friedman (2009), *Iteration Stability for Simple Newtonian Stellar Systems*
[230]: Huang et al. (2008), *Quasiequilibrium models for triaxially deformed rotating compact stars*
[231]: Henriksson et al. (2016), *Initial data for high-compactness black hole–neutron star binaries*

Numerical simulations of thermal noise in thin mirror coatings

5

Publication

This chapter is based on the article *High-accuracy numerical models of Brownian thermal noise in thin mirror coatings* [3], submitted to *Phys. Rev. D* on Nov 16, 2021 (arXiv:2111.06893). It applies my new elliptic solver in `SpECTRE` to simulate thermal noise in interferometric gravitational-wave detectors to unprecedented accuracy. This is possible because the thermal noise problem involves elliptic equations similar to the general-relativistic initial data problems discussed in the preceding Chapter 4. It shows the interdisciplinary application of my work, in the field of gravitational-wave astronomy alone.

Authors **Nils L. Vu**, Samuel Rodriguez, Tom Włodarczyk, Geoffrey Lovelace, Harald P. Pfeiffer, Gabriel S. Bonilla, Nils Deppe, François Hébert, Lawrence E. Kidder, Jordan Moxon, and William Throwe

Abstract Brownian coating thermal noise in detector test masses is limiting the sensitivity of current gravitational-wave detectors on Earth. Therefore, accurate numerical models can inform the ongoing effort to minimize Brownian coating thermal noise in current and future gravitational-wave detectors. Such numerical models typically require significant computational resources and time, and often involve closed-source commercial codes. In contrast, open-source codes give complete visibility and control of the simulated physics and enable direct assessment of the numerical accuracy. In this article, we use the open-source numerical relativity code `SpECTRE` and adopt a novel discontinuous Galerkin numerical method to model Brownian coating thermal noise. We demonstrate that `SpECTRE` achieves significantly higher accuracy than a previous approach at a fraction of the computational cost. Furthermore, we numerically model Brownian coating thermal noise in multiple sub-wavelength crystalline coating layers for the first time. Our new numerical method has the potential to enable fast exploration of realistic mirror configurations, and hence to guide the search for optimal mirror geometries, beam shapes and coating materials for gravitational-wave detectors.

Declaration of authorship This article is a collaborative research effort between the authors. I led the editorial process of writing the article, argued the direction it should take, wrote most of the text, implemented the numerical scheme presented in Section 5.2 in the `SpECTRE` code, performed all numerical simulations presented in Section 5.3, and produced all figures in the article. I also co-supervised Tom Włodarczyk for the duration of his master thesis project, which led to contributions to this article. He implemented the elasticity equations in `SpECTRE`, studied the construction of suitable computational domains, and performed simulations with `SpECTRE` in preparation for our results. Samuel Rodriguez performed the numerical simulations with the `deal.ii` code that are

presented in Figs. 5.2 and 5.3 for comparison, and performed simulations with `SpECTRE` in preparation for our results. Geoffrey Lovelace and Harald Pfeiffer acted as advisors in this project and wrote parts of Sections 5.1 and 5.2.1. The remaining co-authors contributed to the `SpECTRE` code in a manner that enabled the present research. For example, Gabriel Bonilla contributed coordinate maps that are essential to construct the computational domains. The contributions by all authors were essential to conduct the present research, and highlight the collaborative research effort with the open-source `SpECTRE` code. Aside from the contributions listed above, the work presented in this article is my own.

5.1 Introduction

Brownian coating thermal noise is the limiting noise source for current-generation, ground-based gravitational-wave detectors in their most sensitive frequency bands. For instance, following the A+ upgrade anticipated for completion in the mid 2020s, the Laser Intererometer Gravitational-Wave Observatory (LIGO) detector noise is dominated by Brownian coating thermal noise at frequencies $f \sim 100\,\mathrm{Hz}$ [29]. This noise arises from thermal fluctuations in the reflective coatings of the detectors' test masses [232].

[29]: Barsotti et al. (2018), *The A+ design curve*

[232]: Cole et al. (2013), *Tenfold reduction of Brownian noise in high-reflectivity optical coatings*

Therefore, a reduction of the Brownian coating thermal noise directly increases a detector's sensitivity and thus its astronomical reach. Theoretical models of Brownian coating thermal noise are important for working toward this goal. Thermal noise modeling typically follow the approach pioneered by Levin [166], which computes the thermal noise in terms of an auxiliary elasticity calculation using the fluctuation-dissipation theorem [233–235]. While an approximate analytic solution is well known in the limit where coating thickness and edge effects can be neglected, numerical calculations of thermal noise are necessary to study effects that arise from the finite test-mass size, the finite coating thickness, and from crystalline materials.

[166]: Levin (1998), *Internal thermal noise in the LIGO test masses: A Direct approach*

[233]: Callen and Welton (1951), *Irreversibility and generalized noise*
[234]: Bernard and B. Callen (1959), *Irreversible Thermodynamics of Nonlinear Processes and Noise in Driven Systems*
[235]: Kubo (1966), *The fluctuation-dissipation theorem*

In this article we calculate Brownian coating thermal noise by numerically solving the auxiliary linear elasticity problem. Such numerical simulations typically adopt a conventional finite-element approach, as some of the authors did in Ref. [167]. These methods are widely used, but achieving high accuracy with them can require significant computational resources and time, because of their relatively slow rates of convergence.

[167]: Lovelace, Demos, and Khan (2018), *Numerically modeling Brownian thermal noise in amorphous and crystalline thin coatings*

For the first time to our knowledge, we apply a discontinuous Galerkin (DG) method to model Brownian coating thermal noise. DG methods are well suited to this problem because they can retain high-order convergence in the presence of discontinuities, which arise at the interfaces between the mirror substrate and its reflective coatings. In this article, we extend the DG method for elliptic equations presented in Ref. [1] to problems with discontinuous material properties. With this extension, our method converges exponentially with resolution, allowing us to solve coating thermal noise problems numerically at high accuracy using considerably less computational resources and time than conventional finite-element methods.

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

We implement the numerical method and the elastostatic equations in SpECTRE [10], a new open-source numerical relativity code. While SpECTRE's primary aim is to model merging black holes and neutron stars, the elliptic solver needed to construct initial data for such simulations is also very well positioned to solve the DG-discretized elastostatics equations for thermal noise modeling [1, 2]. As an open-source code, our approach has advantages compared to the closed-source and commercial solutions that are often adopted: we can directly control the physics incorporated in the calculation, and we can assess the accuracy and convergence rate of our simulations in a straightforward way. Our code also benefits from SpECTRE's task-based parallelism approach, implemented using the Charm++ [141] library, enabling our code to efficiently scale to large numbers of compute cores [142].

[10]: SpECTRE, spectre-code.org

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.
[2]: **Vu** et al. (2022), *A scalable elliptic solver with task-based parallelism for the SpECTRE code*. Chapter 3 of this thesis.

[141]: The Charm++ Parallel Programming System, https://charm.cs.illinois.edu
[142]: Kidder et al. (2017), *SpECTRE: A Task-based Discontinuous Galerkin Code for Relativistic Astrophysics*

This article is organized as follows. Section 5.2 summarizes the elastic problem to be solved and presents the discontinuous Galerkin numerical method. Section 5.3 presents our results using this method to model thermal noise in cylindrical mirrors with thin coatings. We discuss our results and future work in Section 5.4.

5.2 Methods

In this section, we formulate the auxiliary elasticity problem based on Refs. [166, 167], discretize it with the discontinuous Galerkin scheme developed in Ref. [1], and outline the numerical method we employ to solve the discretized problem with the SpECTRE code [2]. Section 5.2.2 details a novel extension of this method to handle discontinuous material properties at layer interfaces.

[166]: Levin (1998), *Internal thermal noise in the LIGO test masses: A Direct approach*
[167]: Lovelace, Demos, and Khan (2018), *Numerically modeling Brownian thermal noise in amorphous and crystalline thin coatings*

5.2.1 Auxiliary elasticity problem

We consider a gravitational-wave detector that measures the position of a test mass with a laser beam with a Gaussian intensity profile

$$p(r) = \frac{1}{\pi r_0^2} e^{-r^2/r_0^2}. \tag{5.1}$$

Here, r is the cylindrical radial coordinate from the center of the beam with width r_0. The intensity profile is normalized so that

$$\int_0^{2\pi} \mathrm{d}\phi \int_0^{\infty} \mathrm{d}r \; r p(r) = 1. \tag{5.2}$$

The laser beam effectively measures a weighted average q of the displacement Z of the test mass surface,

$$q(t) = \int_0^{2\pi} \mathrm{d}\phi \int_0^{R} \mathrm{d}r \; r p(r,\phi) Z(r,\phi,t). \tag{5.3}$$

As shown by Levin [166], Brownian thermal noise can be calculated from the energy dissipated in an auxiliary elastic problem. Specifically, to compute the thermal noise at frequency f, one applies an oscillating

pressure to the face of the mirror with frequency f, with a pressure distribution profile $p(r)$ equal to the beam intensity, and with an amplitude F_0. In this auxiliary problem, the energy W_{diss} will be dissipated in each cycle of the oscillation. The fluctuation-dissipation theorem relates this dissipated energy W_{diss} to the thermal noise, specifically to the power spectral density S_q associated with q,[1]

$$S_q = \frac{2k_{\text{B}}T}{\pi^2 f^2}\frac{W_{\text{diss}}}{F_0^2}, \tag{5.4}$$

where T is the mirror temperature and k_{B} is Boltzmann's constant. Because $W_{\text{diss}} \propto F_0^2$, it follows that S_q does not depend on the overall amplitude F_0.

1: See, e.g., Eq. (11.90) in Thorne and Blandford [16].

[16]: Thorne and Blandford (2017), *Modern Classical Physics*

For frequencies $f \sim 100\,\text{Hz}$ much lower than the resonant frequencies $f \sim 10^4\text{Hz}$ of the test-mass materials, the dissipated power can be computed using the quasistatic approximation. In this approximation, a static pressure is applied to the mirror with amplitude F_0 and profile $p(r)$, and the dissipated energy can be written as

$$W_{\text{diss}} = U\phi, \tag{5.5}$$

where U is the potential energy stored in the deformation of the test-mass and ϕ is the material's loss angle determined by the material's imaginary, dissipative elastic moduli.

Therefore, our goal in this article is to solve the equations of elastostatics for the deformation of the test mass,

$$\nabla_i T^{ij} = f^j(\boldsymbol{x}), \tag{5.6}$$

when its surface is subjected to an applied pressure with profile $p(r)$. Here, T^{ij} is the *stress* and we adopt the Einstein summation convention so that repeated tensor indices are summed over. The source f^j is the force density acting on each volume element of the mirror as a function of position $\boldsymbol{x}$, which vanishes in our situation, $f^j = 0$. The pressure acting on the external surface of the test-mass will be reflected in suitable boundary conditions.

Equation (5.6) is an equation for the *displacement vector field* $u^i(\boldsymbol{x})$, which describes the deformation of the elastic material as a function of the undeformed coordinates.[2] The symmetric part of the gradient of the displacement vector field is the *strain*

$$S_{kl} = \nabla_{(k} u_{l)}. \tag{5.7}$$

2: I denote the displacement field with $u^i(\boldsymbol{x})$ instead of $\xi^i(\boldsymbol{x})$ in this article to avoid confusion with the logical coordinates $\boldsymbol{\xi}$.

For sufficiently small F_0, the strain is proportional to the applied stress,

$$T^{ij} = -Y^{ijkl} S_{kl}, \tag{5.8}$$

where the *constitutive relation* $Y^{ijkl}(\boldsymbol{x})$ captures the elastic properties of the material in the linear regime. The constitutive relation is symmetric on its first two indices, on its last two indices, and under exchange of the first pair of indices with the second pair of indices.

Inserting Eqs. (5.7) and (5.8) into Eq. (5.6) yields the equations of linear

elasticity,

$$-\nabla_i Y^{ijkl} \nabla_{(k} u_{l)} = f^j(\boldsymbol{x}), \tag{5.9}$$

which we will solve numerically.

We consider layers of piecewise amorphous or cubic-crystalline constitutive relations. The amorphous constitutive relation is isotropic and homogeneous,

$$Y^{ijkl} = \lambda \delta^{ij} \delta^{kl} + \mu \left(\delta^{ik} \delta^{jl} + \delta^{il} \delta^{jk} \right), \tag{5.10}$$

with *Lamé parameter* λ and *shear modulus* μ.[3] A cubic-crystalline material is characterized by the constitutive relation

$$Y^{ijkl} = \begin{cases} c_{11} & \text{for } i = j = k = l \\ c_{12} & \text{for } i = j, k = l, i \neq k \\ c_{44} & \text{for } i = k, j = l, i \neq j \text{ or } i = l, j = k, i \neq j \end{cases} \tag{5.11}$$

where c_{11}, c_{12} and c_{44} are three independent material parameters. The constitutive relation in Eq. (5.9) is composed by discontinuously choosing either Eq. (5.10) or Eq. (5.11) in each layer of the material.

3: The Lamé parameter can also be replaced by the *bulk modulus* $K = \lambda + 2\mu/3$. Alternatively, the two parameters can be replaced by the *Young's modulus* $Y = 9K\mu/(3K + \mu) = \mu(3\lambda + 2\mu)/(\lambda + \mu)$ and the *Poisson ratio* $\sigma = (3K - 2\mu)/(2(3K + \mu)) = \lambda/(2(\lambda + \mu))$.

After solving the linear elasticity equations, Eq. (5.9), the potential energy is evaluated by an integral over the volume of the material,

$$U = -\frac{1}{2} \int_V dV S_{ij} T^{ij}. \tag{5.12}$$

For a material with a thin, reflective coating with different elastic properties than the substrate, the dissipated energy, Eq. (5.5), decomposes as [236]

$$W_{\text{diss}} = U_{\text{sub}}\, \phi_{\text{sub}} + U_{\text{coat}}\, \phi_{\text{coat}}, \tag{5.13}$$

[236]: Harry et al. (2002), *Thermal noise in interferometric gravitational wave detectors due to dielectric optical coatings*

where U_{sub} and ϕ_{sub} are the potential energy and loss angle of the substrate, respectively, while U_{coat} and ϕ_{coat} are the potential energy and the loss angle of the coating. Note that a material can also have different loss angles associated with the different independent elastic moduli of a material [237]. We do not consider further decompositions of the elastic potential energy in this article, but note that such quantities can straightforwardly be extracted from our simulations.

[237]: Hong et al. (2013), *Brownian thermal noise in multilayer coated mirrors*

An approximate analytic solution exists for amorphous materials in the limit where the coating thickness d is small compared to both the size of the mirror and the width r_0 of the pressure profile. The approximate coating thermal noise is [4]

$$\begin{aligned} S_q^{\text{coat}} = & \frac{k_B T}{\pi^2 f} \frac{1 - \sigma_{\text{sub}}^2}{r_0 Y_{\text{sub}}} \frac{d}{r_0} \frac{\phi_{\text{coat}}}{Y_{\text{sub}} Y_{\text{coat}} (1 - \sigma_{\text{coat}}^2)(1 - \sigma_{\text{sub}}^2)} \\ & \times \left(Y_{\text{coat}}^2 (1 + \sigma_{\text{sub}})^2 (1 - 2\sigma_{\text{sub}})^2 + \right. \\ & \left. Y_{\text{sub}}^2 (1 + \sigma_{\text{coat}})^2 (1 - 2\sigma_{\text{coat}}) \right). \end{aligned} \tag{5.14}$$

4: See Eq. (22) in Ref. [236], where $w = \sqrt{2}\, r_0$, $\phi_\parallel = \phi_\phi = \phi_{\text{coat}}$, and we consider only the coating contribution.

5.2.2 Discontinuous Galerkin discretization

We employ the discontinuous Galerkin (DG) scheme detailed in Ref. [1] to

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

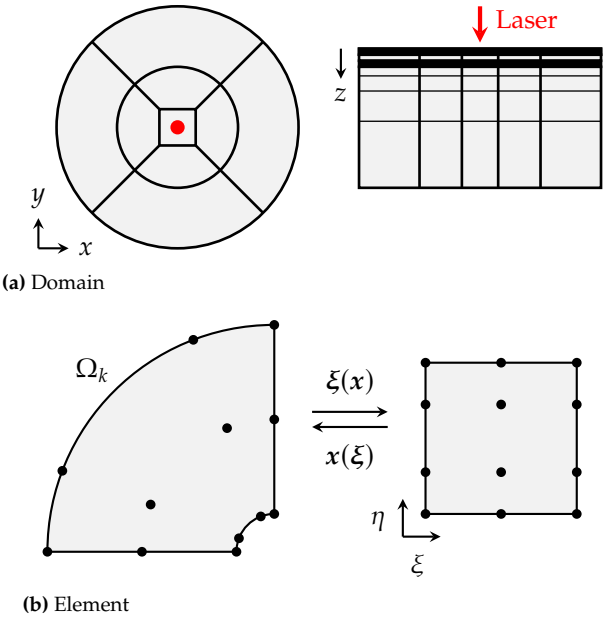


Figure 5.1: *Top:* Geometry of our layered cylindrical domain, with the laser beam indicated in red. Four wedge-shaped elements envelop a cuboid. Another set of wedges extends to the outer radius of the cylinder. In z direction the cylinder is partitioned into layers that can have different material properties (black and gray). The substrate layer has a logarithmic coordinate map in z direction and is split in two twice in this example (thin horizontal lines). *Bottom:* The coordinate transformation $\boldsymbol{\xi}(\boldsymbol{x})$ maps an element to a reference cube $\boldsymbol{\xi} \in [-1,1]^3$ with logical coordinate-axes $\boldsymbol{\xi} = (\xi, \eta, \zeta)$. In this example we chose $N_{k,\xi} = 3$ and $N_{k,\eta} = 4$ LGL collocation points along ξ and η, respectively.

discretize the elasticity problem, Eq. (5.9). We summarize the discretization scheme in this section, and extend it to problems with discontinuous material properties.

Skip ahead

The following two sections summarize Chapter 2 of this thesis. If you have read Chapter 2, skip ahead to the discussion of the numerical flux, Eq. (5.25).

Domain decomposition

We simulate a cylindrical mirror in three dimensions with radius R and height H. The cylinder axis coincides with the z axis of our coordinates, and the plane $z = 0$ represents the surface of the mirror on which the external pressure $p(r)$ is applied. We decompose the cylindrical domain $\Omega = [0, R] \times [0, 2\pi) \times [0, H]$ into a set of nonoverlapping elements $\Omega_k \subset \Omega$ shaped like deformed cubes, as illustrated in Fig. 5.1a (h refinement). Each element carries a coordinate map from the Cartesian coordinates $\boldsymbol{x} \in \Omega_k$, in which the elasticity equations (5.9) are formulated, to *logical* coordinates $\boldsymbol{\xi} \in [-1,1]^3$ representing the reference cube, as illustrated in Fig. 5.1b. The coordinate map to the reference cube is characterized by its Jacobian,

$$\mathrm{J}^i_j = \frac{\partial x^i}{\partial \xi^j} \tag{5.15}$$

with determinant J and inverse $(\mathrm{J}^{-1})^j_i = \partial \xi^j / \partial x^i$. On the reference cube we choose a set of $N_{k,i}$ Legendre-Gauss-Lobatto (LGL) collocation points in each dimension i (p refinement).

Fields are represented numerically by their values at the collocation points. We denote the set of discrete values for the displacement vector

field u^i within an element Ω_k as

$$\underline{u}^{i,(k)} = (u_1^{i,(k)}, \ldots, u_{N_k}^{i,(k)}), \tag{5.16}$$

and the collection of discrete displacement vector field values over *all* elements as $\underline{u}^i$. The values at the collocation points within an element define a three-dimensional Lagrange interpolation,

$$u^{i,(k)}(\boldsymbol{x}) \coloneqq \sum_{p=1}^{N_k} u_p^i \psi_p(\boldsymbol{\xi}(\boldsymbol{x})) \quad \text{with} \quad \boldsymbol{x} \in \Omega_k, \tag{5.17}$$

where the basis functions $\psi_p(\boldsymbol{\xi})$ are products of Lagrange polynomials,

$$\psi_p(\boldsymbol{\xi}) \coloneqq \prod_{i=1}^{3} \ell_{p_i}(\xi^i) \quad \text{with} \quad \boldsymbol{\xi} \in [-1,1]^3, \tag{5.18}$$

based on the collocation points in the three logical directions of the element. Since Eqs. (5.17) and (5.18) are local to each element, fields over the entire domain are discontinuous across element boundaries.

DG residuals

To formulate the elasticity equations in first-order form for the DG discretization, we use the symmetric strain S_{kl} as auxiliary variable. Following Ref. [1], we first compute the discrete auxiliary variables on the computational grid as

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

$$\underline{S}_{kl} = \mathrm{D}_{(k} \cdot \underline{u}_{l)} + \mathrm{L} \cdot ((n_{(k}\underline{u}_{l)})^* - n_{(k}\underline{u}_{l)}), \tag{5.19}$$

where we make use of the discrete differentiation matrix $\mathrm{D}_i \coloneqq \mathrm{M}^{-1}\mathbb{D}_i$, the mass matrix

$$\mathrm{M}_{pq} = \int_{[-1,1]^3} \psi_p(\boldsymbol{\xi})\psi_q(\boldsymbol{\xi})\,\mathrm{J}\,\mathrm{d}^3\xi\,, \tag{5.20}$$

the stiffness matrix

$$\mathbb{D}_{i,pq} = \int_{[-1,1]^3} \psi_p(\boldsymbol{\xi})\,\frac{\partial\psi_q}{\partial\xi^j}(\boldsymbol{\xi})\,(\mathrm{J}^{-1})_i^j\,\mathrm{J}\,\mathrm{d}^3\xi\,, \tag{5.21}$$

the lifting operator

$$\mathbb{L}_{pq} = \int_{[-1,1]^2} \psi_p(\boldsymbol{\xi})\psi_q(\boldsymbol{\xi})\,\mathrm{J}^\Sigma\,\mathrm{d}^2\xi\,, \tag{5.22}$$

and $\mathrm{L} \coloneqq \mathrm{M}^{-1}\mathbb{L}$ on the element Ω_k [1]. The integral in Eq. (5.22) is over the boundary of the element, $\partial\Omega_k$, where n_i is the outward-pointing unit normal one-form and J^Σ is the surface Jacobian. The operation $\cdot$ denotes matrix multiplication with the field values over the computational grid of the element. In a second step, we compute the DG residuals in strong form [1],

$$-\mathbb{D}_i \cdot Y^{ijkl}\underline{S}_{kl} - \mathbb{L} \cdot ((n_i Y^{ijkl}\underline{S}_{kl})^* - n_i Y^{ijkl}\underline{S}_{kl}) = \mathrm{M} \cdot \underline{f}^j, \tag{5.23}$$

which represent the set of algebraic equations for the values $\underline{u}^i$ of the displacement vector field on the computational grid that we solve numerically.

Numerical flux

The quantities $(n_{(k}\underline{u}_{l)})^*$ and $(n_i Y^{ijkl}\underline{S}_{kl})^*$ in Eq. (5.23) denote a numerical flux that couples grid points across nearest-neighbor element boundaries. We employ the generalized internal-penalty numerical flux developed in Ref. [1], with one notable extension. Contrary to Ref. [1] we allow neighboring elements to define different constitutive relations, meaning $Y^{ijkl}(\boldsymbol{x})$ can be double-valued on shared element boundaries.[5] Therefore, we define the quantity

$$Y_*^{ijkl} = \frac{1}{2}\left(Y_{\text{int}}^{ijkl} + Y_{\text{ext}}^{ijkl}\right), \tag{5.24}$$

where "int" denotes the *interior* side of an element's shared boundary with a neighbor, and "ext" denotes the *exterior* side, i.e. the neighbor's side. With this quantity we can define the numerical flux

$$(n_{(k}\underline{u}_{l)})^* = \frac{1}{2}\left[n_{(k}^{\text{int}}\underline{u}_{l)}^{\text{int}} - n_{(k}^{\text{ext}}\underline{u}_{l)}^{\text{ext}}\right], \tag{5.25a}$$

$$\begin{aligned}(n_i Y^{ijkl}\underline{S}_{kl})^* = \frac{1}{2}&\left[n_i^{\text{int}} Y_{\text{int}}^{ijkl}\mathrm{D}_{(k}\cdot\underline{u}_{l)}^{\text{int}} - n_i^{\text{ext}} Y_{\text{ext}}^{ijkl}\mathrm{D}_{(k}\cdot\underline{u}_{l)}^{\text{ext}}\right]\\ &-\sigma\left[n_i^{\text{int}} Y_*^{ijkl} n_{(k}^{\text{int}}\underline{u}_{l)}^{\text{int}} - n_i^{\text{ext}} Y_*^{ijkl} n_{(k}^{\text{ext}}\underline{u}_{l)}^{\text{ext}}\right],\end{aligned} \tag{5.25b}$$

where $n_i^{\text{ext}} = -n_i^{\text{int}}$ for the purpose of this article. Equation (5.25) is the generalized internal-penalty numerical flux defined in Ref. [1], with a choice between Y_{int}^{ijkl}, Y_{ext}^{ijkl}, and Y_*^{ijkl} for every occurrence of the constitutive relation. The particular choice in Eq. (5.25) ensures that the numerical flux remains consistent, meaning that $(n_i Y^{ijkl}\underline{S}_{kl})^* = -n_i^{\text{int}} T^{ij}$ when both $n_i^{\text{int}} Y_{\text{int}}^{ijkl}\mathrm{D}_{(k}\cdot\underline{u}_{l)}^{\text{int}} = -n_i^{\text{ext}} Y_{\text{ext}}^{ijkl}\mathrm{D}_{(k}\cdot\underline{u}_{l)}^{\text{ext}} =: -n_i^{\text{int}} T^{ij}$ and $n_{(k}^{\text{int}}\underline{u}_{l)}^{\text{int}} = -n_{(k}^{\text{ext}}\underline{u}_{l)}^{\text{ext}}$. In particular, note that the penalty term in Eq. (5.25b) vanishes when the displacement is continuous across the boundary, and that the numerical flux admits solutions where the stress is continuous across the boundary but the strain is not. Such solutions may arise in a layered material under stress, because the layers remain "glued together" but each layer responds to the stress differently.

The penalty function in Eq. (5.25b) is

$$\sigma = C\,\frac{(\max(p^{\text{int}}, p^{\text{ext}}) + 1)^2}{\min(h^{\text{int}}, h^{\text{ext}})}, \tag{5.26}$$

where we make use of the polynomial degree p and a measure of the element size, h, orthogonal to the element boundary on either side of the interface, as detailed in Ref. [1]. We choose $C = 100$ in this article.

Boundary conditions

We impose boundary conditions through fluxes, i.e. by a choice of exterior quantities in the numerical flux (5.25). Specifically, on external boundaries

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

5: In the language of Ref. [1] we allow the fluxes $\mathcal{F}_a{}^i[u_A, v_A; \boldsymbol{x}]$ to be double-valued on shared element boundaries.

we set

$$(n_{(k}\underline{u}_{l)})^{\text{ext}} = (n_{(k}\underline{u}_{l)})^{\text{int}} - 2n^{\text{int}}_{(k}\underline{u}^{\text{b}}_{l)} \quad \text{and} \tag{5.27a}$$

$$(n_i Y^{ijkl}\underline{S}_{kl})^{\text{ext}} = (n_i Y^{ijkl}\underline{S}_{kl})^{\text{int}} + 2n_i^{\text{int}}\underline{T}^{ij}_{\text{b}}, \tag{5.27b}$$

where we choose either u^i_{b} to impose Dirichlet boundary conditions, or $n_i^{\text{int}}T^{ij}_{\text{b}}$ to impose Neumann boundary conditions on the boundary collocation points, and set the respective other quantity to its interior value.

For the thermal noise problem we impose the pressure induced by the laser beam,

$$n_i^{\text{int}}T^{ij}_{\text{b}} = n^j p(r), \tag{5.28}$$

as Neumann boundary condition on the $z = 0$ side of the cylindrical mirror, where $p(r)$ is the laser beam profile given in Eq. (5.1). On the side of the mirror facing away from the laser we impose

$$u^i_{\text{b}} = 0 \quad (\text{“fixed”}) \tag{5.29}$$

as Dirichlet boundary condition, and on the mantle we impose

$$n_i^{\text{int}}T^{ij}_{\text{b}} = 0 \quad (\text{“free”}) \tag{5.30}$$

as Neumann boundary condition. Equation (5.29) means that the back of the mirror is held in place, whereas Eq. (5.30) implies no pressure on the sides, which however, are free to deform in response to the pressure applied to the front.

5.2.3 SpECTRE elliptic solver

Once discretized, the linear algebraic equations (5.23) are solved numerically for the displacement vector field values $\underline{u}^i$ on all elements and grid points in the computational domain. As is typical for discretized elliptic equations, Eq. (5.23) defines a matrix equation

$$\mathcal{A}\underline{u} = \underline{b}, \tag{5.31}$$

where $\underline{u}$ denotes the set of all $N_{\text{DOF}} = 3\,N_{\text{points}} = 3\,\sum_k N_k$ displacement vector field values in the computational domain, and $\mathcal{A}$ is a matrix with $N_{\text{DOF}} \times N_{\text{DOF}}$ entries. To solve Eq. (5.31) means inverting the matrix $\mathcal{A}$. However, as the resolution of the computational domain increases, the matrix $\mathcal{A}$ easily becomes too large to construct explicitly, to store on an ordinary computer, and to invert directly.

Therefore, we solve Eq. (5.31) with the elliptic solver component of the open-source `SpECTRE` code [2, 10]. It employs an iterative generalized minimal residual (GMRES) algorithm to solve Eq. (5.31) to the requested precision. A multigrid preconditioner accelerates the GMRES algorithm by supporting each iteration with an approximate solution from a hierarchy of successively coarser grids. On every grid, an additive Schwarz smoother decomposes the problem into many overlapping subproblems, one per element in the domain, which are solved independently and in parallel. The subproblems are distributed across the cores of a computing

[2]: **Vu** et al. (2022), *A scalable elliptic solver with task-based parallelism for the `SpECTRE` code*. Chapter 3 of this thesis.
[10]: SpECTRE, spectre-code.org

cluster by a task-based parallelization paradigm. The elliptic solver is detailed in Ref. [2].

5.3 Results

Our simulations with `SpECTRE` were performed on one or more 16-core compute nodes, each with 64 GB of memory and two eight-core Intel Haswell E5-2630v3 processors clocked at 2.40 GHz, connected with an Intel Omni-Path network. We distribute the elements that compose the computational domain evenly among cores, leaving one core per node free to perform communications.

We compare our results to previous work using an open-source finite element code to calculate the Brownian coating thermal noise for amorphous and crystalline materials. Its methods are described in Secs. 2.4–2.6 of Ref. [167]. The code was built using the `deal.ii` [238, 239] finite element framework and we henceforth refer to it as `deal.ii`. It adopted a standard weak form of the elastostatic equations, discretized them using a conventional finite element approach, and solved them using `deal.ii` with the `PETSc` [136] conjugate gradient linear solver and the `ParaSAILS` preconditioner in the `hypre` [138] package. The `deal.ii` code relies on the Message Passing Interface (MPI) for parallelization.

[167]: Lovelace, Demos, and Khan (2018), *Numerically modeling Brownian thermal noise in amorphous and crystalline thin coatings*

[238]: Arndt et al. (2021), *The `deal.II` Library*

[239]: Arndt et al. (2021), *The deal.II finite element library: Design, features, and insights*

[136]: PETSc, `https://www.mcs.anl.gov/petsc`

[138]: Falgout, Jones, and Yang (2006), *The Design and Implementation of hypre, a Library of Parallel High Performance Preconditioners*

5.3.1 Single-coating comparison

First, we consider the single-coating scenario investigated in Ref. [167] and demonstrate the superior performance of our new approach. We choose the parameters listed in Ref. [167], Table 1 for a cylindrical mirror of radius $R = 12.5\,\text{mm}$ with a single $d = 6.83\,\mu\text{m}$ thin effective-isotropic AlGaAs coating. We simulate the scenario both with the `deal.ii` approach employed in Ref. [167] and with our new approach with the `SpECTRE` code.

Figure 5.2 presents the numerical precision and computational cost of both approaches. To assess the numerical precision we successively increase the resolution in both codes. In `SpECTRE` we increase the resolution by incrementing the number of grid points in all dimensions of all elements in the domain by one, and in `deal.ii` we employ an adaptive mesh-refinement scheme [167]. We compute the error in the elastic potential energy relative to a high-resolution reference simulation. We use a reference configuration simulated in `SpECTRE` where we have split all elements in two along all three dimensions, relative to the highest-resolution configuration included in Fig. 5.2.

We find that both codes converge to the same solution, but our new approach in `SpECTRE` achieves about four orders of magnitude higher accuracy than the `deal.ii` approach using the same number of grid points. Furthermore, our new approach simulates this scenario with sub-percent error in only 30 s on 15 cores, for which the `deal.ii` approach required multiple hours on 324 cores. Our new approach also achieves a fractional error below 10^{-5} in only half a core-hour, or two minutes of real time, which was prohibitively expensive with the `deal.ii` approach.

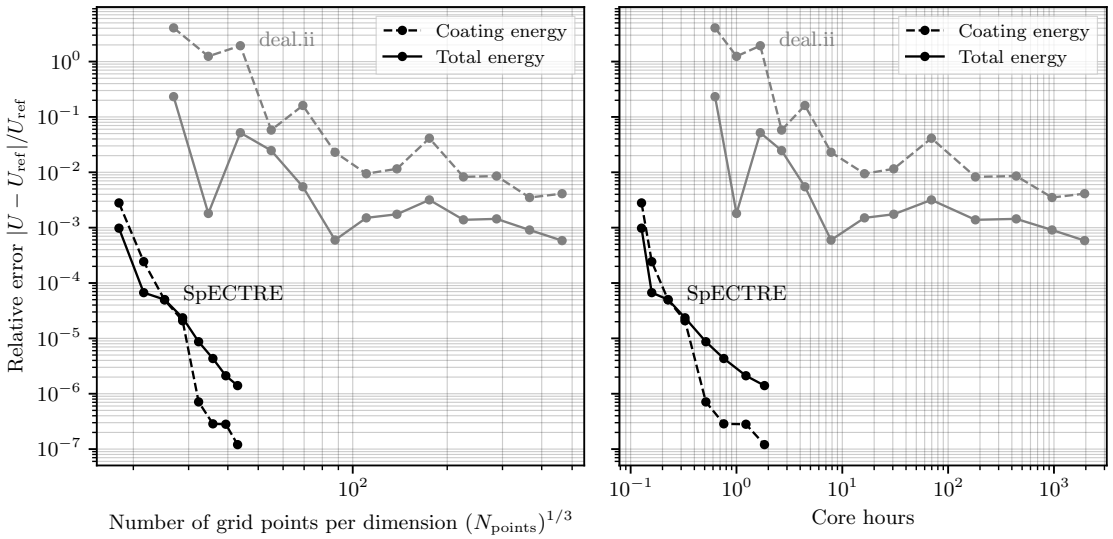


Figure 5.2: Relative error of the potential energy in a single amorphous coating layer (dashed lines) and in the full mirror (solid lines). *Left:* `SpECTRE`, with our discontinuous Galerkin method, resolves the coating layer at high precision using only a fraction of the number of grid points needed by the `deal.ii` approach. *Right:* `SpECTRE` solves the elliptic problem using only a fraction of the computational resources needed by the `deal.ii` approach.

5.3.2 Accuracy of the approximate analytic solution

Second, we study the accuracy of the approximate analytic solution for the single-coating thermal noise, Eq. (5.14), using the superior numerical precision we can now achieve over the results presented in Ref. [167]. The approximate solution holds for a thin coating, $d/r_0 \ll 1$, a semi-infinite mirror, $r_0/R \ll 1$ and $d/R \ll 1$, and for isotropic-homogeneous materials. Therefore, it does not capture the finite-size effects included in our simulations, and approximates the crystalline AlGaAs coating as an amorphous material.

[167]: Lovelace, Demos, and Khan (2018), *Numerically modeling Brownian thermal noise in amorphous and crystalline thin coatings*

To assess the magnitude of the finite-size effects, we employ the simulations detailed in Section 5.3.1, which use the same effective-isotropic model for the AlGaAs coating that underpins the approximate analytic solution. Figure 5.3 presents both the thermal noise computed from the simulations and the approximate analytic solution (black). Error bars are computed as $\Delta\sqrt{S_q^{\text{coat}}}/\sqrt{S_q^{\text{coat}}} = 1/2\,\Delta U_{\text{coat}}/U_{\text{coat}}$ from the relative numerical error in the elastic potential energy. While Ref. [167] estimated the magnitude of finite-size effects for this problem to 7 %, we can now report that their simulations captured the effect to (7.5 ± 0.2) %. With our new numerical method, we can make this statement more precise and report a finite-size effect of (7.616 649 ± 0.000 006) %.

To assess the magnitude of the amorphous approximation to the crystalline coating material, we repeat the simulations with a crystalline constitutive relation. The thermal noise computed from these simulations is presented in Fig. 5.3 as well (red). We refine the estimate of 4 % from Ref. [167] to (4.5 ± 0.2) %, and report (4.667 990 ± 0.000 006) % using our new numerical method.

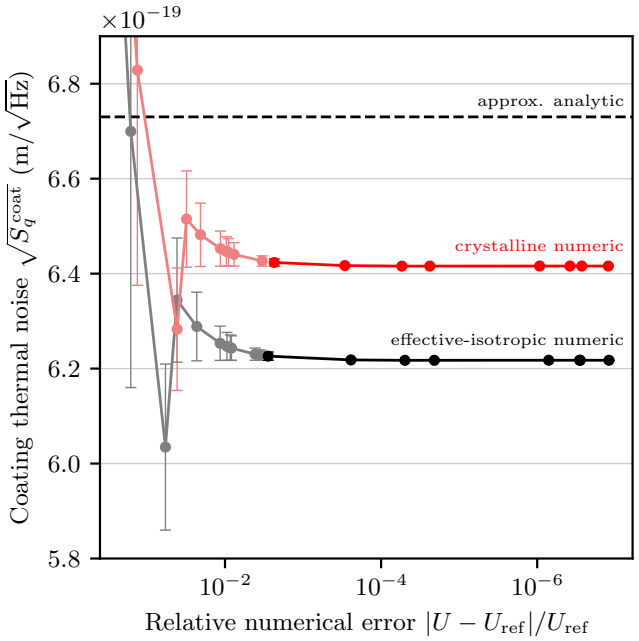


Figure 5.3: Thermal noise in an AlGaAs-coated mirror computed from the approximate analytic solution (5.14) and from our numerical simulations. The effective-isotropic simulation (black) retains the amorphous approximation for the material, but includes finite-size effects. The crystalline simulation (red) eliminates this approximation. Previous simulations with the `deal.ii` approach are shown in lighter colors to the left.

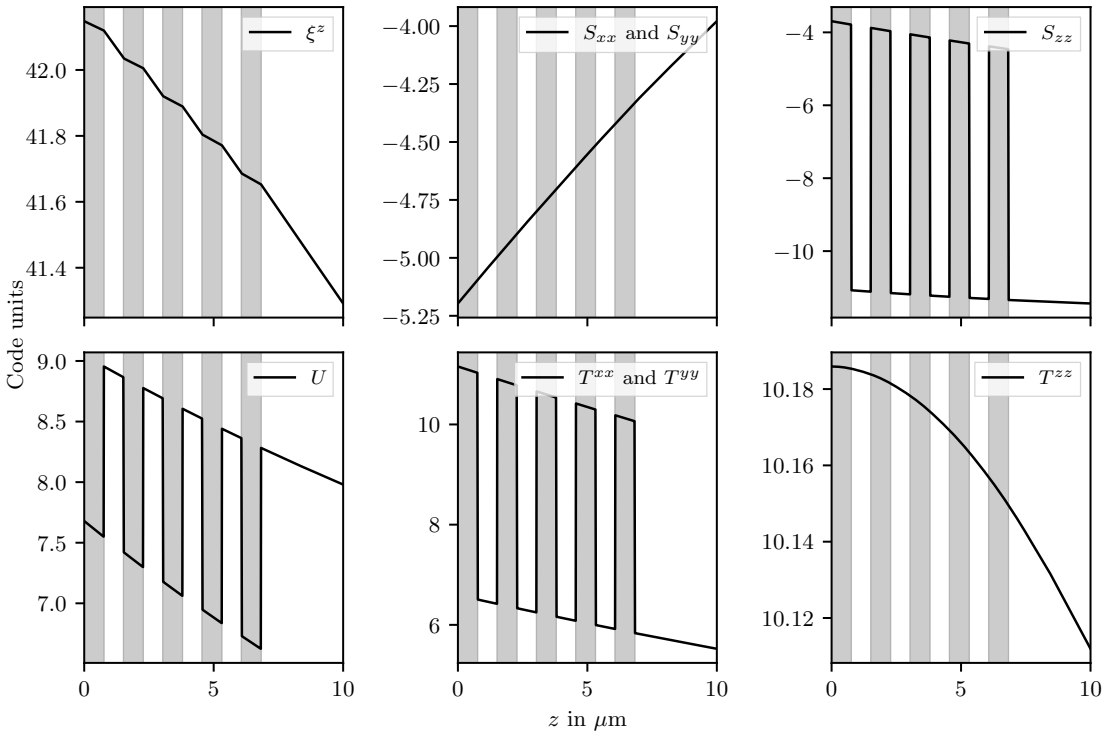


Figure 5.4: Elastic variables along the z-axis for a mirror with multiple thin coating layers, simulated with `SpECTRE`. Gray layers are crystalline AlGaAs, and white regions are fused silica. The nine coating layers have a combined thickness of 6.83 µm and the material extends to $z = 12.5$ mm.

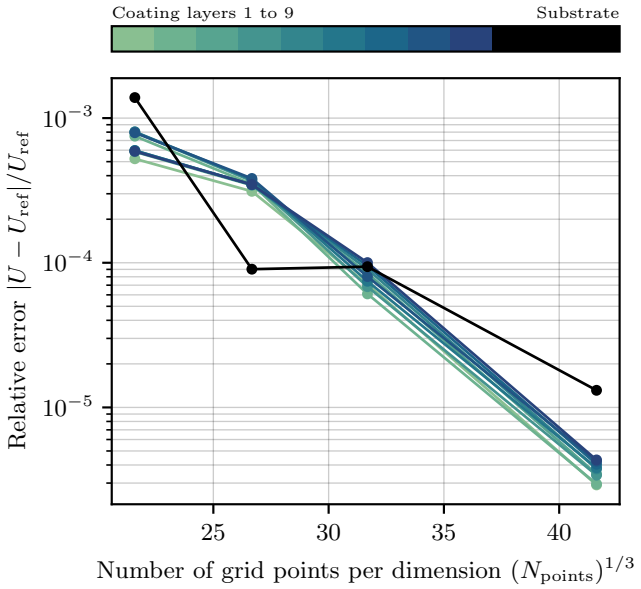


Figure 5.5: Convergence test of the elastic potential energy in each coating layer, and in the substrate. Our new numerical method achieves exponential convergence despite the discontinuous material properties.

5.3.3 Multiple sub-wavelength crystalline coatings

Finally, we apply our new computational approach to a scenario that presents many of the challenges we expect for applications to realistic mirror configurations. We simulate a cylindrical mirror of the same radius $R = 12.5\,\text{mm}$ as before, but split the $d = 6.83\,\mu\text{m}$ thin coating into nine layers, so the thickness of each coating layer is below the typical $1\,\mu\text{m}$ wavelength of the laser. The coating layers alternate between fused silica and crystalline AlGaAs, with the elastic moduli c_{11}, c_{12} and c_{44} listed in Ref. [167], Table 1. Neither sub-wavelength coatings nor multiple layers were simulated in Ref. [167], but our new computational approach in `SpECTRE` achieves both.

[167]: Lovelace, Demos, and Khan (2018), *Numerically modeling Brownian thermal noise in amorphous and crystalline thin coatings*

Figure 5.4 presents our numerical solution of this scenario. Our new computational approach based on discontinuous Galerkin methods resolves the thin coating layers at high accuracy without spurious oscillations. Figure 5.5 presents the numerical precision of the solution. We increase the resolution by incrementing the number of grid points per element and dimension and compute the relative error to a high-resolution reference configuration, as we did in Section 5.3.1. The error converges exponentially, which is a feature of our discontinuous Galerkin method with grid boundaries placed at the layer interfaces.

5.4 Discussion

We have presented a new numerical method to model Brownian thermal noise in thin mirror coatings based on a discontinuous Galerkin (DG) discretization. With our new method, we model thermal noise in a one-inch cylindrical mirror with a microns-thick coating at unprecedented accuracy at a fraction of the time needed in a previous, conventional finite-element approach [167]. Using these high-accuracy simulations, we find that a commonly-used approximate analytic solution overestimates the coating thermal noise for this problem by 7.6 % when taking

only finite-size effects into account, and by 4.7 % when modeling it as a crystalline material, which refines a previous estimate in Ref. [167]. We also demonstrate that, unlike the approach in Ref. [167], our new method is capable of resolving multiple sub-wavelength coatings, including coatings of a cubic-crystalline material. Our new numerical method is implemented in the open-source `SpECTRE` code and the results presented in this article are reproducible with the supplemental input-file configurations.

We found that it is crucial for the success of our new method that the interfaces between layers of different materials coincide with element boundaries in our computational domain. Then, our discontinuous Galerkin discretization with a suitable choice of numerical flux converges exponentially, achieving high accuracy with a small number of grid points. The scheme can potentially be improved in future work. Most notably, an adaptive mesh-refinement (AMR) algorithm would have great potential to further improve the accuracy and efficiency of the scheme, by distributing the resolution in the computational domain to regions and dimensions where it is most needed.

Furthermore, the elliptic solver in the `SpECTRE` code that we employ to solve the discretized problem numerically can be improved to accelerate thermal-noise calculations. The calculations we have presented in this article require a few hundred solver-iterations to converge, or up to ~ 1400 for our highest-resolution simulation with multiple sub-wavelength crystalline coatings. While simple configurations complete in seconds or minutes of real-time on 15 cores, where the previous approach needed hours on 324 cores, the more challenging configurations, which were prohibitively expensive with the previous approach, solve in about an hour on 45 cores.

We expect additional speedup with further improvements to the elliptic solver algorithm in `SpECTRE`. In particular, improvements to its multigrid preconditioner have great potential to speed up the simulations. The multigrid algorithm relies on solving the problem approximately on coarser grids to resolve large-scale modes in the solution. It currently cannot coarsen the grid any further than the size of each coating layer because the layers define the material properties. To accelerate the calculations, we intend to let the multigrid algorithm combine layers with different materials into fiducial coarse layers with effective material properties. This approach is possible because the partitioning of the domain into layers is necessary only to define material properties, not to define the geometry of the domain. Reference [2] shows that the multigrid algorithm can achieve resolution-independent iteration counts when the domain can be coarsened sufficiently. Note that the fiducial coarse layers affect only the convergence behavior of the solver, but have no effect on the solution once the solver has converged.

[2]: **Vu** et al. (2022), *A scalable elliptic solver with task-based parallelism for the `SpECTRE` code*. Chapter 3 of this thesis.

Our numerical models of thermal noise have the potential to inform upgrades that increase the sensitivity of gravitational-wave detectors, using the advanced computational technology that we develop for numerical-relativity simulations in the `SpECTRE` code. In the future, we intend to apply our new numerical method to simulate Brownian thermal noise in more realistic mirror configurations and materials that are under consideration for current and future gravitational-wave detectors, such

as the optimized configuration found in Ref. [240]. While approximate analytic solutions can provide useful estimates, only numerical models can precisely quantify the finite-size effects of changing the mirror geometry. In particular, finite-size effects are more important for real gravitational-wave detectors than for tabletop experiments measuring thermal noise. Tabletop experiments often use small beam sizes to enlarge the thermal noise and hence make it easier to measure, whereas gravitational-wave detectors prefer large beam sizes to minimize thermal noise. Therefore, we plan to employ our new numerical method to explore realistic mirror configurations, with the goal of finding configurations that minimize Brownian coating thermal noise.

[240]: Venugopalan, Arai, and Adhikari (2021), *Global optimization of multilayer dielectric coatings for precision measurements*

Acknowledgments

The authors thank Josh Smith for helpful discussions. Computations were performed with the `SpECTRE` [10] and `deal.ii` [238, 239] codes on the `Minerva` cluster at the Max Planck Institute for Gravitational Physics and on the `Ocean` cluster at Fullerton. The figures in this article were produced with `dgpy` [11], `matplotlib` [172, 173], `TikZ` [174] and `ParaView` [175]. This work was supported in part by the Sherman Fairchild Foundation, by NSF Grants No. PHY-1654359 and No. AST-1559694 at Cal State Fullerton, by NSF Grants No. PHY-2011961, No. PHY-2011968, and No. OAC-1931266 at Caltech and by NSF Grants No. PHY-1912081 and No. OAC-1931280 at Cornell.

[10]: SpECTRE, spectre-code.org

[238]: Arndt et al. (2021), *The `deal.II` Library*

[239]: Arndt et al. (2021), *The deal.II finite element library: Design, features, and insights*

[11]: dgpy, 10.5281/zenodo.5086181

[172]: Hunter (2007), *Matplotlib: A 2D graphics environment*

[173]: matplotlib, 10.5281/zenodo.4268928

[174]: pgf - A Portable Graphic Format for TeX, `github:pgf-tikz/pgf`

[175]: Ahrens, Geveci, and Law (2005), *`ParaView`: An End-User Tool for Large-Data Visualization*

6 Conclusion

In this thesis I have developed an elliptic solver for numerical relativity that scales effectively to supercomputers. It is targeted to support numerical simulations of astrophysical scenarios with the new SpECTRE code [10] for the emerging era of gravitational-wave and multimessenger astronomy. It employs a discontinuous Galerkin (DG) numerical scheme to discretize the elliptic equations, and task-based parallel algorithms to distribute computation to supercomputers.

[10]: SpECTRE, spectre-code.org

The DG scheme that I have developed is applicable to a wide range of nonlinear elliptic problems that appear in numerical relativity, such as the XCTS and puncture initial data formalisms [1]. It also applies to elliptic equations throughout computational physics, such as elasticity problems. It recovers exponential convergence for smooth problems. At the same time, it allows for hp refinement to adapt the grid to the problem at hand. This makes it particularly suited for problems involving discontinuities, such as neutron stars with phase transitions or layered elastic materials, recovering exponential convergence when the discontinuities are placed at grid boundaries. The DG scheme generally needs more grid points than comparable domains in SpEC [55] because points on element boundaries are duplicate, and because we tend to split domains into a larger number of elements to favor parallelization. For instance, where SpEC always employs eleven large subdomains for a binary black hole (BBH) initial data problem, and hence scales to at most eleven cores, my new DG scheme splits the domain into a few hundred smaller elements to balance high-order convergence with parallelizability. Optimal domain decompositions and hp-adaptive mesh refinement (AMR) procedures are major subjects of future research to improve the efficiency of the DG scheme.

[1]: **Fischer** and Pfeiffer (2022), *Unified discontinuous Galerkin scheme for a large class of elliptic equations*. Chapter 2 of this thesis.

[55]: Spectral Einstein Code (SpEC), black-holes.org/code/SpEC

I have developed a stack of task-based parallel iterative algorithms to solve the sparse matrix equations arising from the DG discretization [2]. These algorithms constitute the elliptic solver of the SpECTRE code. The centerpiece is a highly parallelizable multigrid-Schwarz preconditioner supporting a Newton-Krylov algorithm. From an algorithmic perspective, the preconditioner achieves scale-independent iteration counts: the computational work required to solve problems at higher resolution manifests in harder and more numerous subproblems, rather than increasing the number of iterations. From a technological perspective, these subproblems represent tasks that can be distributed on computing clusters. The additive Schwarz algorithm overlaps the computation of subproblems as much as possible by avoiding global synchronization points. Each subproblem represents a preconditioned iterative solve by itself, and I have developed efficient subdomain preconditioners to accelerate the elliptic solver. It has proven capable of distributing over 200 million degrees of freedom effectively to thousands of cores. In the future we intend to explore the performance of the elliptic solver on even larger problems, e.g., involving neutron stars, and more cores.

[2]: **Vu** et al. (2022), *A scalable elliptic solver with task-based parallelism for the SpECTRE code*. Chapter 3 of this thesis.

I have applied the new elliptic solver to generate initial data for numeri-

cal relativity simulations. Using the superposed Kerr-Schild formalism for the XCTS equations, I have demonstrated that the elliptic solver already generates BBH initial data about ten times faster than `SpEC` by parallelizing computation to 120 cores. This means in particular that my new elliptic solver can use modern computing clusters effectively, which consist of nodes with considerably more than the eleven cores usable by the `SpEC` elliptic solver. The feature set of the initial data solver is already comparable to `SpEC`, supporting superposed Schwarzschild and Kerr solutions in various coordinate systems to construct conformal backgrounds, as well as horizon-conforming excision surfaces and negative-expansion boundary conditions for unequal-mass and spinning black hole binaries. The initial data sets are ready to be evolved by the hyperbolic component of the `SpECTRE` code, which is currently completing work on control systems for the position and shape of the excision surfaces during the evolution. Future work on the initial data solver includes rootfinding routines that invoke the elliptic solver repeatedly to control free parameters.

A major avenue for applications of the elliptic solver is initial data with neutron stars. I have demonstrated that the DG scheme converges exponentially for single and binary TOV stars when the stellar surface is placed at grid boundaries. Therefore, I intend to employ the technology used to deform black-hole excision surfaces to conform the grid to stellar surfaces and other discontinuities in the problem, such as phase transitions within the stars. I have demonstrated the capability to solve for the gravity sector of head-on binary neutron star (BNS) initial data based on the density and pressure profiles of two TOV stars at rest. Solving the equations of hydrostatic equilibrium alongside the gravity sector will enable orbiting and spinning BNS initial data. Iteration procedures are used throughout the literature to couple the gravity and hydrostatic sectors and to control free parameters, but they are often fragile and strongly damped. Therefore, I intend to explore solving the coupled XCTS and hydrostatic equations together to obtain more accurate iterations, accelerate the generation of BNS and BHNS initial data, and cover their parameter space robustly.

As an interdisciplinary application of the elliptic solver I have simulated Brownian thermal noise in thin mirror coatings [3]. This source of noise limits the sensitivity of current interferometric gravitational-wave detectors. The DG scheme is able to resolve the thin coating layers with high-order convergence, and the elliptic solver in `SpECTRE` solves these thermal-noise problems orders of magnitude faster than a previous study based on lower-order finite-element methods. Equipped with this capability we have simulated, for the first time, thermal noise in multiple sub-wavelength crystalline coating layers. I intend to extend our fast, high-resolution numerical simulations of thermal noise to mirror configurations under consideration for future upgrades of the LIGO detectors, such as the optimized configuration presented in Ref. [240]. Therefore, our numerical models of thermal noise have the potential to inform upgrades that increase the sensitivity of gravitational-wave detectors, using the advanced computational technology that we develop for numerical-relativity simulations.

[3]: **Vu** et al. (2021), *High-accuracy numerical models of Brownian thermal noise in thin mirror coatings*. Chapter 5 of this thesis.

[240]: Venugopalan, Arai, and Adhikari (2021), *Global optimization of multilayer dielectric coatings for precision measurements*

A fast elliptic solver also has the intriguing potential to support evolutions. For example, the study of planetesimal formation in protoplanetary disks

requires extensive numerical simulations, in particular when magnetic fields are involved to study magneto-rotational or gravitational instabilities [241]. These simulations account for the self-gravity of the disk by solving for its Newtonian gravitational potential alongside the evolution [212–214]. With SpECTRE we can potentially achieve high-resolution simulations of self-gravitating protoplanetary disks using my fast, iterative and task-based parallel elliptic solver for the Newtonian gravity sector, in combination with AMR, load-balancing and our high-order numerical methods for hydrodynamic evolutions.

[241]: Riols and Latter (2019), *Gravitoturbulent dynamos in astrophysical discs*

[212]: Deng, Mayer, and Latter (2020), *Global Simulations of Self-gravitating Magnetized Protoplanetary Disks*
[213]: Hopkins (2015), *A new class of accurate, mesh-free hydrodynamic simulation methods*
[214]: Enzo (2014), *Enzo: An Adaptive Mesh Refinement Code for Astrophysics*

A particularly promising approach to support evolutions within numerical relativity is to stabilize challenging simulations by solving elliptic constraints, such as magnetic divergence cleaning [120], favorable gauges, or even (parts of) the Einstein constraints on a path toward fully constrained evolutions [85]. Constraint damping and divergence cleaning schemes are essential for contemporary numerical simulations, and it remains to be seen to what extent they can be supported by a fast elliptic solver. Along a similar line of thought, stiff terms in evolution equations can potentially be replaced by elliptic problems in implicit-explicit (IMEX) evolution schemes [120–123]. To solve these elliptic problems alongside an evolution we can explore dispatching only a few asynchronous Schwarz-smoothing steps every couple of time steps to keep constraint violations below a tolerance with maximal parallel efficiency, or run full multigrid-Schwarz preconditioned Newton-Krylov iterations when needed. Applications like these have received little attention in the numerical relativity community in the last decade, but may be enabled by a fast and highly parallel elliptic solver.

[85]: Cheong, Lin, and Li (2020), *Gmunu: Toward multigrid based Einstein field equations solver for general-relativistic hydrodynamics simulations*

[120]: Cheong et al. (2021), *An extension of Gmunu: General-relativistic resistive magnetohydrodynamics based on staggered-meshed constrained transport with elliptic cleaning*
[121]: Pareschi and Russo (2005), *Implicit-explicit runge-kutta schemes and applications to hyperbolic systems with relaxation*
[122]: Lau, Lovelace, and Pfeiffer (2011), *Implicit-explicit (IMEX) evolution of single black holes*
[123]: Ripperda et al. (2019), *General relativistic resistive magnetohydrodynamics with robust primitive variable recovery for accretion disk simulations*

Appendix

A

Visualizations of gravitational-wave events

Figures 1.3 and 1.7 to 1.9 show snapshots from visualizations of gravitational-wave events that I have produced for LIGO outreach efforts. They show three "exceptional" events that were detected by the LIGO and Virgo collaborations during the first part of the third observing run (O3a):

- `GW190412` [35] (Figs. 1.3 and 1.7 and visualization [242]) was the first observed black hole merger with significantly unequal masses (about $30\,\mathrm{M}_\odot$ and $8\,\mathrm{M}_\odot$), a nonzero spin of the larger black hole, and an indication that the system was precessing. My visualization is based on a precessing `SpEC` simulation with $q = 3.5$, $\chi_1 = (0.52, 0, 0.3)$, and $\chi_2 = 0$.[1]
- `GW190814` [36] (Fig. 1.8 and visualization [243]) was a merger event with particularly unequal masses (about $23\,\mathrm{M}_\odot$ and $2.5\,\mathrm{M}_\odot$ to $3\,\mathrm{M}_\odot$). Since the mass of the smaller object was so small, it may have been either a light black hole or a heavy neutron star. My visualization assumes it was a black hole. The large mass ratio emphasized higher harmonics in the gravitational-wave signal, and kept the signal in band of the detectors for over 10 s (or about 160 orbits). Since this inspiral is longer than any available NR simulation with consistent parameters, I combined the `SpEC` simulation `SXS:BBH:1108` [9] ($q = 9.2$ and nonspinning) with a long waveform computed by the `SEOBNRv4HM` model [46]. I highlight the higher harmonics in the visualization.
- `GW190521` [37] (Fig. 1.9 and visualization [244]) was a very short signal from the merger of two particularly heavy black holes (about $85\,\mathrm{M}_\odot$ and $66\,\mathrm{M}_\odot$). For numerical relativity simulations, only the mass *ratio* of the two black holes is relevant, not their total mass. For the visualization I used the `SpEC` simulation `SXS:BBH:1006` [9] with $q = 1.03$, $\chi_1 = (0.64, 0.21, -0.35)$, and $\chi_2 = (-0.48, 0.18, 0.50)$.

I also contributed to the visualization of the BHNS merger event `GW200115` [245].

The pictures were produced with my `gwpv` software package [12]. It dispatches to the `ParaView` scientific visualization toolkit [175] to generate three-dimensional renderings of gravitational waveforms from a numerical simulation or from a waveform model. Waveforms are typically stored in the form of complex spin-weighted spherical harmonic (SWSH) modes $h_{lm}(t)$. Currently, only the SXS waveform file format is supported, which stores the SWSH modes extrapolated from a `SpEC` simulation in an HDF5 file [9]. For the visualization, I load the waveform data and evaluate the SWSH expansion (1.10) for the GW strain $rh(t, r, \theta, \phi)$ in a three-dimensional volume. I evaluate the expansion on a uniform grid of points, since I have found that `ParaView` provides the best volume

[35]: LIGO, Virgo (2020), *GW190412: Observation of a Binary-Black-Hole Coalescence with Asymmetric Masses*

[242]: **Fischer** et al. (2020), *Visualization of the GW190412 event*

1: The `SpEC` simulation is not yet public at the time of writing. It is identified internally by the name `PrecBBH000029`, and will carry the public name `SXS:BBH:1253`.

[36]: LIGO, Virgo (2020), *GW190814: Gravitational Waves from the Coalescence of a 23 Solar Mass Black Hole with a 2.6 Solar Mass Compact Object*

[243]: **Fischer** et al. (2020), *Visualization of the GW190814 event*

[9]: SXS (2019), *The SXS collaboration catalog of binary black hole simulations*

[46]: Cotesta et al. (2018), *Enriching the Symphony of Gravitational Waves from Binary Black Holes by Tuning Higher Harmonics*

[37]: LIGO, Virgo (2020), *GW190521: A Binary Black Hole Merger with a Total Mass of* $150M_\odot$

[244]: **Fischer**, Pfeiffer, and Buonanno (2020), *Visualization of the GW190521 event*

[245]: Dietrich et al. (2022), *Simulation of an NSBH coalescence consistent with GW200115*

[12]: gwpv, `github:nilsvu/gwpv`

[175]: Ahrens, Geveci, and Law (2005), *`ParaView`: An End-User Tool for Large-Data Visualization*

[9]: SXS (2019), *The SXS collaboration catalog of binary black hole simulations*

(1.10): $h = h_+ - ih_\times = \frac{1}{r}\sum_{l=2}^{\infty}\sum_{m=-l}^{l} h_{lm}(t)\,{}_{-2}Y_{lm}(\theta,\phi)$

rendering capabilities on uniform grids. The waveform modes $h_{lm}(t)$ are evaluated at the retarded time $t_r = t - r/\lambda$ on the grid, where r is the Euclidean coordinate distance from the origin and λ is a scale factor used to control the visualization. Since the waveform modes are stored at a set of discrete sample times, their evaluation at the retarded times requires an interpolation, which I have optimized for uniformly sampled waveforms. The polarization of the visualized gravitational radiation is selected by taking either the real part or the imaginary part of the SWSH expansion (1.10). Optionally, the radial scaling with $1/r$ is either added or omitted. At an inner radius $r_{\min}$ and at an outer radius $r_{\max}$ the visualized quantity is multiplied by a smoothstep function to activate and deactivate the visualization radially.

[246]: Python package `spherical`, 10.5281/zenodo.4045222

I employ the `spherical` package [246] to compute the spin-weighted spherical harmonics ${}_{-2}Y_{lm}(\theta, \phi)$ in the expansion (1.10) on every grid point. Evaluating the SWSH grid incurs a considerable computational cost, but it can be cached and reused for all timesteps in a visualization. I also cache the SWSH grids on disk and reuse them for multiple visualization. The SWSH caching is essential to achieve reasonable rendering performance.

The number of modes that contribute to the SWSH expansion (1.10) is configurable. In many cases, only the dominant $(2, 2)$ mode is needed for the visualization, but higher modes become important at high mass ratios or for asymmetric GW events such as hyperbolic encounters. For the `GW190814` visualization [243] I also rendered images of individual modes, scaled in amplitude to their respective maxima.

[243]: **Fischer** et al. (2020), *Visualization of the GW190814 event*

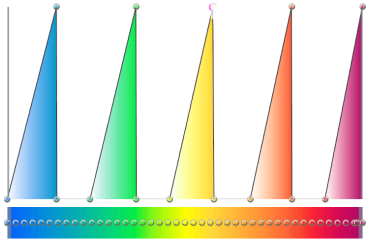


Figure A.1: A "sawtooth" transfer function in `ParaView`. The abscissa defines the color map and the ordinate defines the opacity for mapping the GW strain in the volume to colors.

Once the GW strain is computed on the uniform grid, a `ParaView` pipeline produces the image. The pipeline dispatches to `ParaView`'s GPU-based volume renderer, which traces rays through the volume to accumulate the color for each pixel. The GW strain is translated to colors by a *transfer function* (Fig. A.1). I generally construct the transfer function by choosing a colormap, and by dividing a range of values for the GW strain into a number of uniformly or logarithmically spaced peaks in opacity ("sawtooth"-shaped opacity function). This choice of transfer function creates visually pleasing sharp contours of different colors that "peel off" from the center.

In addition to the gravitational radiation, the visualizations show the apparent horizons extracted from the numerical simulations along with optional visual indicators such as the trajectories, spins, and two-dimensional Ricci scalar of the horizons. Visualizations of the apparent horizons are particularly instructive near merger, where they deform considerably and form a common horizon. Additional `ParaView` filters can be added to the visualizations, e.g., to create slices or clips, or for camera motion. All components of the visualizations are composed in *scenes*, represented by a stack of YAML configuration files.

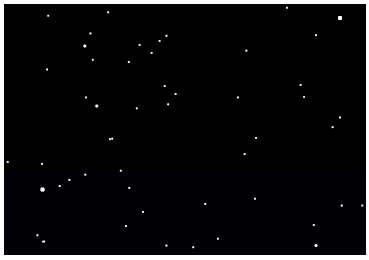


Figure A.2: Background with somewhat realistically distributed stars.

The background of the visualizations shows randomly distributed stars (Fig. A.2). To generate the background image, I uniformly sample the positions of the stars on the sky, and sample random distances D in the volume, i.e. from a cubic power law. From the randomly sampled distances I scale the radius of each star by $1/D$ according to the small-angle approximation, and compute the opacity of each star according

to its flux, i.e., scaled by $1/D^2$. The background image is mapped to a sphere in `ParaView` that envelops the visualization volume.

The visualizations were rendered on the GPU machine `Slartibartfast` at AEI Potsdam with an Nvidia Quadro P4000 GPU. Rendering a single frame at high resolution took ~40 s and required up to ~40 GB of RAM and ~1 GB of GPU memory. Since rendering multiple frames is "embarassingly parallel" I employ Python `multiprocessing` to distribute the rendering to multiple cores. The number of cores that the rendering can scale to is typically limited by the available memory on a node.

Since the visualizations use only waveform data (and optionally apparent horizons from a numerical simulation), they can be generated from any waveform published in a catalog or produced by a waveform model. In future work I intend to add optional support for visualizing the strong-field gravitational field extracted from a numerical simulation to capture the violent dynamics near merger. I also intend to visualize hydrodynamic quantities extracted from simulations involving neutron stars. Finally, a particularly intriguing prospect is to extend the open-source ray tracing capabilities of `ParaView` to support gravitational lensing. Work along this avenue has been done by Bohn et al. [247] in `SpEC`. Building gravitational lensing directly into `ParaView` would benefit from its advanced rendering capabilities, and has the potential to generate quite spectacular visualizations of numerical relativity simulations.

[247]: Bohn et al. (2015), *What does a binary black hole merger look like?*

B

Input-file configurations

These input files reproduce the results referenced throughout this thesis. They all run with the `SolveXcts` executable of the SpECTRE code [10], compiled on the following branch:

- `https://github.com/nilsvu/spectre/tree/thesis`

B.1 `KerrSchildDefect.yaml`

```yaml
Background: &kerr
  KerrSchild:
    Mass: 1.
    Spin: [0., 0., 0.]
    Center: [0., 0., 0.]

InitialGuess: Flatness

DomainCreator:
  Shell:
    InnerRadius: 2.
    OuterRadius: 10.
    InitialRefinement: 1
    InitialGridPoints: [6, 6]
    UseEquiangularMap: True
    EquatorialCompression: None
    WhichWedges: All
    RadialPartitioning: []
    RadialDistribution: [Logarithmic]
    TimeDependence: None
    Shape: None
    BoundaryConditions:
      InnerBoundary:
        ApparentHorizon:
          Center: [0., 0., 0.]
          Rotation: [0., 0., 0.]
          # Neumann-type lapse boundary condition, Eq. (4.2)
          Lapse: Auto
          # Dirichlet-type lapse boundary condition, Eq. (2.53b)
          # Lapse: *kerr
          NegativeExpansion: None
      OuterBoundary:
        AnalyticSolution:
          ConformalFactor: Dirichlet
          LapseTimesConformalFactor: Dirichlet
          ShiftExcess: Dirichlet

Discretization:
  DiscontinuousGalerkin:
    PenaltyParameter: 1.
    Massive: True

Observers:
  VolumeFileName: "KerrSchildVolume"
  ReductionFileName: "KerrSchildReductions"

NonlinearSolver:
  NewtonRaphson:
    ConvergenceCriteria:
      # Increase MaxIterations to ~20 for Fig. 4.2
      MaxIterations: 1
      RelativeResidual: 0.
      AbsoluteResidual: 1.e-10
    SufficientDecrease: 1.e-4
    MaxGlobalizationSteps: 40
    Damping: None
    Verbosity: Quiet

LinearSolver:
  Gmres:
    ConvergenceCriteria:
      MaxIterations: 30
      RelativeResidual: 1.e-4
      AbsoluteResidual: 1.e-12
    Verbosity: Quiet

  Multigrid:
    Iterations: 1
    MaxLevels: Auto
    PreSmoothing: True
    PostSmoothingAtBottom: False
    Verbosity: Silent
    OutputVolumeData: False
    ElementAllocation:
      WeightByNumPoints: True

  SchwarzSmoother:
    Iterations: 3
    MaxOverlap: 2
    Verbosity: Silent
    SubdomainSolver:
      Gmres:
        ConvergenceCriteria:
          MaxIterations: 3
          RelativeResidual: 1.e-4
          AbsoluteResidual: 1.e-12
        Verbosity: Silent
        Restart: None
        Preconditioner:
          MinusLaplacian:
            Solver:
              ExplicitInverse:
                FillFactor: 1
                Verbosity: Silent
            BoundaryConditions: Auto
    SkipResets: True
    ObservePerCoreReductions: False

EventsAndTriggers:
  ? Always
  : - ObserveNorms:
        SubfileName: ErrorNorms
        TensorsToObserve:
          - Name: Error(ConformalFactor)
            NormType: L2Norm
            Components: Sum
```

```yaml
          - Name: Error(LapseTimesConformalFactor)
            NormType: L2Norm
            Components: Sum
          - Name: Error(ShiftExcess)
            NormType: L2Norm
            Components: Sum
    - ObserveNorms:
        SubfileName: Norms
        TensorsToObserve:
          - Name: HamiltonianConstraint
            NormType: L2Norm
            Components: Individual
          - Name: MomentumConstraint
            NormType: L2Norm
            Components: Individual
    - ObserveFields:
        SubfileName: VolumeData
        VariablesToObserve:
          - ConformalFactor
          - LapseTimesConformalFactor
          - ShiftExcess
          - Error(ConformalFactor)
          - Error(LapseTimesConformalFactor)
          - Error(ShiftExcess)
          - HamiltonianConstraint
          - MomentumConstraint
        InterpolateToMesh: None
        CoordinatesFloatingPointType: Float
        FloatingPointTypes: [Float]
```

B.2 KerrSchildSpin.yaml

```yaml
Background: &kerr
  KerrSchild:
    Mass: &kerr_mass 1.
    Spin: &kerr_spin [0., 0., 0.9]
    Center: [0., 0., 0.]

InitialGuess: *kerr

DomainCreator:
  Shell:
    # $\mathring{r}_{\min} = r_+ = M + \sqrt{M^2 - a^2}$, Eq. (4.4)
    # Set to $\mathring{r}_{\min} = M = 1$ for Fig. 4.6
    InnerRadius: 1.4358898943540672
    OuterRadius: 10.
    Shape:
      Mass: *kerr_mass
      Spin: *kerr_spin
      Modes: [20, 20]
    RadialPartitioning: []
    RadialDistribution: [Logarithmic]
    InitialRefinement: {{L}}
    InitialGridPoints: [{{P+1}}, {{P+1}}]
    UseEquiangularMap: True
    EquatorialCompression: None
    WhichWedges: All
    TimeDependence: None
    BoundaryConditions:
      InnerBoundary:
        ApparentHorizon:
          Center: [0., 0., 0.]
          # $\Omega_r = \chi/(2\mathring{r}_{\min})$, Eq. (4.11)
          # Set to `[0., 0., -0.45]` for Fig. 4.6
          Rotation: [0., 0., -0.3133945031366293]
          Lapse: *kerr
          # Set to `*kerr` for Fig. 4.6
          NegativeExpansion: None
      OuterBoundary:
        AnalyticSolution:
          ConformalFactor: Dirichlet
          LapseTimesConformalFactor: Dirichlet
          ShiftExcess: Dirichlet

Discretization:
  DiscontinuousGalerkin:
    PenaltyParameter: 1.
    Massive: True

Observers:
  VolumeFileName: "KerrSchildVolume"
  ReductionFileName: "KerrSchildReductions"

NonlinearSolver:
  NewtonRaphson:
    ConvergenceCriteria:
      MaxIterations: 20
      RelativeResidual: 0.
      AbsoluteResidual: 1.e-10
    SufficientDecrease: 1.e-4
    MaxGlobalizationSteps: 40
    Damping: None
    Verbosity: Verbose

LinearSolver:
  Gmres:
    ConvergenceCriteria:
      MaxIterations: 100
      RelativeResidual: 1.e-3
      AbsoluteResidual: 1.e-11
    Verbosity: Quiet

  Multigrid:
    Iterations: 1
    MaxLevels: Auto
    PreSmoothing: True
    PostSmoothingAtBottom: False
    Verbosity: Silent
    OutputVolumeData: False
    ElementAllocation:
      WeightByNumPoints: True

  SchwarzSmoother:
    MaxOverlap: 2
    Iterations: 3
    Verbosity: Silent
    SubdomainSolver:
      Gmres:
        ConvergenceCriteria:
          MaxIterations: 3
          RelativeResidual: 1.e-4
          AbsoluteResidual: 1.e-10
        Verbosity: Silent
        Restart: None
        Preconditioner:
          MinusLaplacian:
            Solver:
              ExplicitInverse:
                FillFactor: 1
                Verbosity: Silent
            BoundaryConditions: Auto
    ObservePerCoreReductions: False
    SkipResets: True

EventsAndTriggers:
  ? Always
  : - ObserveNorms:
        SubfileName: ErrorNorms
        TensorsToObserve:
          - Name: Error(ConformalFactor)
            NormType: L2Norm
            Components: Individual
          - Name: Error(ConformalFactor)
            NormType: L2IntegralNorm
            Components: Individual
          - Name: Error(LapseTimesConformalFactor)
```

```
            NormType: L2Norm
            Components: Individual
          - Name: Error(LapseTimesConformalFactor)
            NormType: L2IntegralNorm
            Components: Individual
          - Name: Error(ShiftExcess)
            NormType: L2Norm
            Components: Individual
          - Name: Error(ShiftExcess)
            NormType: L2IntegralNorm
            Components: Individual
    - ObserveNorms:
        SubfileName: Norms
        TensorsToObserve:
          - Name: ConformalFactor
            NormType: Max
            Components: Individual
          - Name: Lapse
            NormType: Min
            Components: Individual
          - Name: Magnitude(ShiftExcess)
            NormType: Max
            Components: Individual
          - Name: HamiltonianConstraint
            NormType: L2Norm
            Components: Individual
          - Name: HamiltonianConstraint
            NormType: L2IntegralNorm
            Components: Individual
          - Name: MomentumConstraint
            NormType: L2Norm
            Components: Individual
          - Name: MomentumConstraint
            NormType: L2IntegralNorm
            Components: Individual
    - ObserveFields:
        SubfileName: VolumeData
        VariablesToObserve:
          - ConformalFactor
          - Error(ConformalFactor)
          - Error(LapseTimesConformalFactor)
          - Error(ShiftExcess)
          - Lapse
          - Shift
          - Magnitude(ShiftExcess)
          - SpatialMetric
          - ExtrinsicCurvature
          - HamiltonianConstraint
          - MomentumConstraint
        InterpolateToMesh: None
        CoordinatesFloatingPointType: Double
        FloatingPointTypes: [Double]
  # For Fig. 4.6:
  - ObserveAtPoint:
        Coordinates: [1.6, 0, 0]
        SubfileName: NearBulge
        TensorsToObserve:
          - ConformalFactor
          - LapseTimesConformalFactor
          - ShiftExcess
          - HamiltonianConstraint
          - MomentumConstraint
  - ObserveAtPoint:
        Coordinates: [0, 0, 1.2]
        SubfileName: NearTop
        TensorsToObserve:
          - ConformalFactor
          - LapseTimesConformalFactor
          - ShiftExcess
          - HamiltonianConstraint
          - MomentumConstraint
  - ObserveAtPoint:
        Coordinates: [9, 0, 0]
        SubfileName: Outside
        TensorsToObserve:
          - ConformalFactor
          - LapseTimesConformalFactor
          - ShiftExcess
          - HamiltonianConstraint
          - MomentumConstraint
```

B.3 BbhKsi.yaml

```
Background: &background
  Binary:
    XCoords: [&x_left -5., &x_right 5.]
    ObjectA: &kerr_left
      Schwarzschild:
        Mass: 1.
        Coordinates: KerrSchildIsotropic
    ObjectB: &kerr_right
      Schwarzschild:
        Mass: 1.
        Coordinates: KerrSchildIsotropic
    AngularVelocity: 0.04266
    Expansion: 0.
    FalloffWidths: None

InitialGuess: *background

DomainCreator:
  BinaryCompactObject:
    ObjectA:
      InnerRadius: 1.2727410334221052
      OuterRadius: 3.5
      XCoord: *x_left
      Interior:
        ExciseWithBoundaryCondition:
          ApparentHorizon:
            Center: [*x_left, 0., 0.]
            Rotation: [0., 0., 0.]
            Lapse: Auto
            NegativeExpansion: None
      Shape: None
      UseLogarithmicMap: True
    ObjectB:
      InnerRadius: 1.2727410334221052
      OuterRadius: 3.5
      XCoord: *x_right
      Interior:
        ExciseWithBoundaryCondition:
          ApparentHorizon:
            Center: [*x_right, 0., 0.]
            Rotation: [0., 0., 0.]
            Lapse: Auto
            NegativeExpansion: None
      Shape: None
      UseLogarithmicMap: True
    UseEquiangularMap: True
    EnvelopingCube:
      Radius: 55.
      UseProjectiveMap: True
      Sphericity: 0.
    OuterShell:
      InnerRadius: 60.
      OuterRadius: 1e6
      RadialDistribution: Inverse
      BoundaryCondition: Flatness
    InitialRefinement:
      ObjectAShell:              [{{L+1}}, {{L+1}}, {{L+1}}]
      ObjectBShell:              [{{L+1}}, {{L+1}}, {{L+1}}]
      ObjectACube:               [{{L+1}}, {{L+1}}, {{L  }}]
      ObjectBCube:               [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeUpperZLeft:  [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeLowerZLeft:  [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeUpperYLeft:  [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeLowerYLeft:  [{{L+1}}, {{L+1}}, {{L  }}]
```

```
      EnvelopingCubeLowerX:      [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeUpperZRight: [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeLowerZRight: [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeUpperYRight: [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeLowerYRight: [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeUpperX:      [{{L+1}}, {{L+1}}, {{L  }}]
      CubedShellUpperZLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellLowerZLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellUpperYLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellLowerYLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellLowerX:          [{{L+1}}, {{L+1}}, {{L  }}]
      CubedShellUpperZRight:     [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellLowerZRight:     [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellUpperYRight:     [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellLowerYRight:     [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellUpperX:          [{{L+1}}, {{L+1}}, {{L  }}]
      OuterShellUpperZLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerZLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellUpperYLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerYLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerX:          [{{L+1}}, {{L+1}}, {{L  }}]
      OuterShellUpperZRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerZRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellUpperYRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerYRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellUpperX:          [{{L+1}}, {{L+1}}, {{L  }}]
    InitialGridPoints:
      ObjectAShell:   [{{P+1}}, {{P+1}}, {{P+4}}]
      ObjectBShell:   [{{P+1}}, {{P+1}}, {{P+4}}]
      ObjectACube:    [{{P+1}}, {{P+1}}, {{P+2}}]
      ObjectBCube:    [{{P+1}}, {{P+1}}, {{P+2}}]
      EnvelopingCube: [{{P+1}}, {{P+1}}, {{P+1}}]
      CubedShell:     [{{P+1}}, {{P+1}}, {{P+1}}]
      OuterShell:     [{{P+1}}, {{P+1}}, {{P  }}]

Discretization:
  DiscontinuousGalerkin:
    PenaltyParameter: 1.
    Massive: True

Observers:
  VolumeFileName: "BbhVolume"
  ReductionFileName: "BbhReductions"

NonlinearSolver:
  NewtonRaphson:
    ConvergenceCriteria:
      MaxIterations: 20
      RelativeResidual: 0.
      AbsoluteResidual: 1.e-10
    SufficientDecrease: 1.e-4
    MaxGlobalizationSteps: 40
    Damping: None
    Verbosity: Verbose

LinearSolver:
  Gmres:
    ConvergenceCriteria:
      MaxIterations: 100
      RelativeResidual: 1.e-3
      AbsoluteResidual: 1.e-10
    Verbosity: Quiet

  Multigrid:
    Iterations: 1
    MaxLevels: Auto
    PreSmoothing: True
    PostSmoothingAtBottom: True
    Verbosity: Silent
    OutputVolumeData: False
    ElementAllocation:
      WeightByNumPoints: True

  SchwarzSmoother:
    MaxOverlap: 2
    Iterations: 3
    Verbosity: Silent
    SubdomainSolver:
      Gmres:
        ConvergenceCriteria:
          MaxIterations: 3
          RelativeResidual: 1.e-4
          AbsoluteResidual: 1.e-10
        Verbosity: Silent
        Restart: None
        Preconditioner:
          MinusLaplacian:
            Solver:
              ExplicitInverse:
                FillFactor: 1
                Verbosity: Silent
            BoundaryConditions: Auto
    SkipResets: True
    ObservePerCoreReductions: False

EventsAndTriggers:
  ? HasConverged
  : - ObserveNorms:
        SubfileName: Norms
        TensorsToObserve:
          - Name: ConformalFactor
            NormType: Max
            Components: Individual
          - Name: Lapse
            NormType: Min
            Components: Individual
          - Name: Magnitude(ShiftExcess)
            NormType: Max
            Components: Individual
          - Name: HamiltonianConstraint
            NormType: L2Norm
            Components: Individual
          - Name: MomentumConstraint
            NormType: L2Norm
            Components: Individual
    - ObserveFields:
        SubfileName: VolumeData
        VariablesToObserve:
          - ConformalFactor
          - Lapse
          - Shift
          - ShiftExcess
          - Magnitude(ShiftExcess)
          - HamiltonianConstraint
          - MomentumConstraint
        InterpolateToMesh: None
        CoordinatesFloatingPointType: Float
        FloatingPointTypes: [Float]
```

B.4 BbhSks.yaml

```
Background: &background
  Binary:
    XCoords: [&x_left -8., &x_right 8.]
    ObjectA: &kerr_left
      KerrSchild:
        Mass: 0.4229
        Spin: [0., 0., 0.]
        Center: [0., 0., 0.]
    ObjectB: &kerr_right
      KerrSchild:
        Mass: 0.4229
        Spin: [0., 0., 0.]
        Center: [0., 0., 0.]
    AngularVelocity: 0.0144
    Expansion: 0.
    FalloffWidths: [4.8, 4.8]
```

```yaml

InitialGuess: *background

DomainCreator:
  BinaryCompactObject:
    ObjectA:
      InnerRadius: 0.8458
      OuterRadius: 5.
      XCoord: *x_left
      Interior:
        ExciseWithBoundaryCondition:
          ApparentHorizon:
            Center: [*x_left, 0., 0.]
            Rotation: [0., 0., 0.]
            Lapse: *kerr_left
            NegativeExpansion: None
      Shape: None
      UseLogarithmicMap: True
    ObjectB:
      InnerRadius: 0.8458
      OuterRadius: 5.
      XCoord: *x_right
      Interior:
        ExciseWithBoundaryCondition:
          ApparentHorizon:
            Center: [*x_right, 0., 0.]
            Rotation: [0., 0., 0.]
            Lapse: *kerr_right
            NegativeExpansion: None
      Shape: None
      UseLogarithmicMap: True
    UseEquiangularMap: False
    EnvelopingCube:
      Radius: 55.
      UseProjectiveMap: True
      Sphericity: 0.
    OuterShell:
      InnerRadius: 60.
      OuterRadius: 300.
      RadialDistribution: Inverse
      BoundaryCondition: Flatness
    InitialRefinement:
      ObjectAShell:              [{{L+1}}, {{L+1}}, {{L+1}}]
      ObjectBShell:              [{{L+1}}, {{L+1}}, {{L+1}}]
      ObjectACube:               [{{L+1}}, {{L+1}}, {{L  }}]
      ObjectBCube:               [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeUpperZLeft:  [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeLowerZLeft:  [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeUpperYLeft:  [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeLowerYLeft:  [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeLowerX:      [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeUpperZRight: [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeLowerZRight: [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeUpperYRight: [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeLowerYRight: [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeUpperX:      [{{L+1}}, {{L+1}}, {{L  }}]
      CubedShellUpperZLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellLowerZLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellUpperYLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellLowerYLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellLowerX:          [{{L+1}}, {{L+1}}, {{L  }}]
      CubedShellUpperZRight:     [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellLowerZRight:     [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellUpperYRight:     [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellLowerYRight:     [{{L  }}, {{L+1}}, {{L  }}]
      CubedShellUpperX:          [{{L+1}}, {{L+1}}, {{L  }}]
      OuterShellUpperZLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerZLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellUpperYLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerYLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerX:          [{{L+1}}, {{L+1}}, {{L  }}]
      OuterShellUpperZRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerZRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellUpperYRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerYRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellUpperX:          [{{L+1}}, {{L+1}}, {{L  }}]
    InitialGridPoints:
      ObjectAShell:   [{{P+1}}, {{P+1}}, {{P+4}}]
      ObjectBShell:   [{{P+1}}, {{P+1}}, {{P+4}}]
      ObjectACube:    [{{P+1}}, {{P+1}}, {{P+2}}]
      ObjectBCube:    [{{P+1}}, {{P+1}}, {{P+2}}]
      EnvelopingCube: [{{P+1}}, {{P+1}}, {{P+1}}]
      CubedShell:     [{{P+1}}, {{P+1}}, {{P+1}}]
      OuterShell:     [{{P+1}}, {{P+1}}, {{P  }}]

Discretization:
  DiscontinuousGalerkin:
    PenaltyParameter: 1.
    Massive: True

Observers:
  VolumeFileName: "BbhVolume"
  ReductionFileName: "BbhReductions"

NonlinearSolver:
  NewtonRaphson:
    ConvergenceCriteria:
      MaxIterations: 20
      RelativeResidual: 0.
      AbsoluteResidual: 1.e-10
    SufficientDecrease: 1.e-4
    MaxGlobalizationSteps: 40
    Damping: None
    Verbosity: Verbose

LinearSolver:
  Gmres:
    ConvergenceCriteria:
      MaxIterations: 100
      RelativeResidual: 1.e-3
      AbsoluteResidual: 1.e-10
    Verbosity: Quiet

  Multigrid:
    Iterations: 1
    MaxLevels: Auto
    PreSmoothing: True
    PostSmoothingAtBottom: True
    Verbosity: Silent
    OutputVolumeData: False
    ElementAllocation:
      WeightByNumPoints: True

  SchwarzSmoother:
    MaxOverlap: 2
    Iterations: 3
    Verbosity: Silent
    SubdomainSolver:
      Gmres:
        ConvergenceCriteria:
          MaxIterations: 3
          RelativeResidual: 1.e-4
          AbsoluteResidual: 1.e-10
        Verbosity: Silent
        Restart: None
        Preconditioner:
          MinusLaplacian:
            Solver:
              ExplicitInverse:
                FillFactor: 1
                Verbosity: Silent
            BoundaryConditions: Auto
    SkipResets: True
    ObservePerCoreReductions: False

EventsAndTriggers:
  ? HasConverged
  : - ObserveAtPoint:
        Coordinates: [0, 0, 0]
        SubfileName: Origin
        TensorsToObserve:
          - ConformalFactor
          - LapseTimesConformalFactor
          - ShiftExcess
    - ObserveAtPoint:
        Coordinates: [8.846, 0, 0]
        SubfileName: NearHorizon
        TensorsToObserve:
          - ConformalFactor
          - LapseTimesConformalFactor
```

```
          - ShiftExcess
    - ObserveAtPoint:
        Coordinates: [100, 0, 0]
        SubfileName: FarField
        TensorsToObserve:
          - ConformalFactor
          - LapseTimesConformalFactor
          - ShiftExcess
    - ObserveNorms:
        SubfileName: Norms
        TensorsToObserve:
          - Name: ConformalFactor
            NormType: Max
            Components: Individual
          - Name: Lapse
            NormType: Min
            Components: Individual
          - Name: Magnitude(ShiftExcess)
            NormType: Max
            Components: Individual
          - Name: HamiltonianConstraint
            NormType: L2Norm
            Components: Individual
          - Name: MomentumConstraint
            NormType: L2Norm
            Components: Individual
    - ObserveFields:
        SubfileName: VolumeData
        VariablesToObserve:
          - ConformalFactor
          - Lapse
          - Shift
          - ShiftExcess
          - Magnitude(ShiftExcess)
          - HamiltonianConstraint
          - MomentumConstraint
        InterpolateToMesh: None
        CoordinatesFloatingPointType: Float
        FloatingPointTypes: [Float]
```

B.5 BbhDomain.yaml

```
Background: &background
  Binary:
    XCoords: [&x_left -8., &x_right 8.]
    ObjectA: &kerr_left
      KerrSchild:
        Mass: 0.4229
        Spin: [0., 0., 0.]
        Center: [0., 0., 0.]
    ObjectB: &kerr_right
      KerrSchild:
        Mass: 0.4229
        Spin: [0., 0., 0.]
        Center: [0., 0., 0.]
    AngularVelocity: 0.0144
    Expansion: 0.
    FalloffWidths: [4.8, 4.8]

InitialGuess: *background

DomainCreator:
  BinaryCompactObject:
    ObjectA:
      InnerRadius: 0.8458
      OuterRadius: 4.
      XCoord: *x_left
      Interior:
        ExciseWithBoundaryCondition:
          ApparentHorizon:
            Center: [*x_left, 0., 0.]
            Rotation: [0., 0., 0.]
            Lapse: *kerr_left
            NegativeExpansion: None
      Shape: None
      UseLogarithmicMap: True
    ObjectB:
      InnerRadius: 0.8458
      OuterRadius: 4.
      XCoord: *x_right
      Interior:
        ExciseWithBoundaryCondition:
          ApparentHorizon:
            Center: [*x_right, 0., 0.]
            Rotation: [0., 0., 0.]
            Lapse: *kerr_right
            NegativeExpansion: None
      Shape: None
      UseLogarithmicMap: True
    UseEquiangularMap: True
    EnvelopingCube:
      Radius: 60.
      UseProjectiveMap: True
      Sphericity: 1.
    OuterShell:
      InnerRadius: Auto
      OuterRadius: 300.
      RadialDistribution: Inverse
      BoundaryCondition: Flatness
    InitialRefinement:
      ObjectAShell:              [{{L+1}}, {{L+1}}, {{L+1}}]
      ObjectBShell:              [{{L+1}}, {{L+1}}, {{L+1}}]
      ObjectACube:               [{{L+1}}, {{L+1}}, {{L+1}}]
      ObjectBCube:               [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeUpperZLeft:  [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeLowerZLeft:  [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeUpperYLeft:  [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeLowerYLeft:  [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeLowerX:      [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeUpperZRight: [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeLowerZRight: [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeUpperYRight: [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeLowerYRight: [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeUpperX:      [{{L+1}}, {{L+1}}, {{L+1}}]
      OuterShellUpperZLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerZLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellUpperYLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerYLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerX:          [{{L+1}}, {{L+1}}, {{L  }}]
      OuterShellUpperZRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerZRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellUpperYRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerYRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellUpperX:          [{{L+1}}, {{L+1}}, {{L  }}]
    InitialGridPoints:
      ObjectAShell:   [{{P+1}}, {{P+1}}, {{P+5}}]
      ObjectBShell:   [{{P+1}}, {{P+1}}, {{P+5}}]
      ObjectACube:    [{{P+1}}, {{P+1}}, {{P+2}}]
      ObjectBCube:    [{{P+1}}, {{P+1}}, {{P+2}}]
      EnvelopingCube: [{{P+1}}, {{P+1}}, {{P+1}}]
      OuterShell:     [{{P+1}}, {{P+1}}, {{P+1}}]

Discretization:
  DiscontinuousGalerkin:
    PenaltyParameter: 1.
    Massive: True

Observers:
  VolumeFileName: "BbhVolume"
  ReductionFileName: "BbhReductions"

NonlinearSolver:
  NewtonRaphson:
    ConvergenceCriteria:
      MaxIterations: 20
      RelativeResidual: 1.e-10
```

```yaml
      AbsoluteResidual: 1.e-11
    SufficientDecrease: 1.e-4
    MaxGlobalizationSteps: 40
    Damping: None
    Verbosity: Verbose

LinearSolver:
  Gmres:
    ConvergenceCriteria:
      MaxIterations: 100
      RelativeResidual: 1.e-3
      AbsoluteResidual: 1.e-10
    Verbosity: Quiet

  Multigrid:
    Iterations: 1
    MaxLevels: Auto
    PreSmoothing: True
    PostSmoothingAtBottom: True
    Verbosity: Silent
    OutputVolumeData: False
    ElementAllocation:
      WeightByNumPoints: True

  SchwarzSmoother:
    MaxOverlap: 2
    Iterations: 3
    Verbosity: Silent
    SubdomainSolver:
      Gmres:
        ConvergenceCriteria:
          MaxIterations: 3
          RelativeResidual: 1.e-4
          AbsoluteResidual: 1.e-10
        Verbosity: Silent
        Restart: None
        Preconditioner:
          MinusLaplacian:
            Solver:
              ExplicitInverse:
                FillFactor: 1
                Verbosity: Silent
            BoundaryConditions: Auto
    SkipResets: True
    ObservePerCoreReductions: False

EventsAndTriggers:
  ? HasConverged
  : - ObserveAtPoint:
        Coordinates: [0, 0, 0]
        SubfileName: Origin
      TensorsToObserve:
        - ConformalFactor
        - LapseTimesConformalFactor
        - ShiftExcess
  - ObserveAtPoint:
      Coordinates: [8.846, 0, 0]
      SubfileName: NearHorizon
      TensorsToObserve:
        - ConformalFactor
        - LapseTimesConformalFactor
        - ShiftExcess
  - ObserveAtPoint:
      Coordinates: [100, 0, 0]
      SubfileName: FarField
      TensorsToObserve:
        - ConformalFactor
        - LapseTimesConformalFactor
        - ShiftExcess
  - ObserveNorms:
      SubfileName: Norms
      TensorsToObserve:
        - Name: ConformalFactor
          NormType: Max
          Components: Individual
        - Name: Lapse
          NormType: Min
          Components: Individual
        - Name: Magnitude(ShiftExcess)
          NormType: Max
          Components: Individual
        - Name: HamiltonianConstraint
          NormType: L2Norm
          Components: Individual
        - Name: MomentumConstraint
          NormType: L2Norm
          Components: Individual
  - ObserveFields:
      SubfileName: VolumeData
      VariablesToObserve:
        - ConformalFactor
        - LapseTimesConformalFactor
        - Lapse
        - Shift
        - ShiftExcess
        - Magnitude(ShiftExcess)
        - HamiltonianConstraint
        - MomentumConstraint
      InterpolateToMesh: None
      CoordinatesFloatingPointType: Double
      FloatingPointTypes: [Double]
```

B.6 BbhSpin.yaml

```yaml
Background: &background
  Binary:
    XCoords: [&x_left -3.87, &x_right 11.61]
    ObjectA: &kerr_left
      KerrSchild:
        Mass: &mass_left 0.75
        Spin: &spin_left [0., 0.49, -0.755]  # $|\chi| \approx 0.9$
        Center: [0., 0., 0.]
    ObjectB: &kerr_right
      KerrSchild:
        Mass: &mass_right 0.25
        Spin: &spin_right [0., 0., 0.]
        Center: [0., 0., 0.]
    AngularVelocity: 0.01515
    Expansion: 0.
    FalloffWidths: [7.5, 2.5]

InitialGuess: *background

DomainCreator:
  BinaryCompactObject:
    ObjectA:
      # $r_+ = 1.0768098644472042$, $\mathring{r}_{\min} \approx 0.9\, r_+$
      InnerRadius: 0.97
      OuterRadius: 4.
      XCoord: *x_left
      Shape:
        Mass: *mass_left
        Spin: *spin_left
        Modes: [20, 20]
      Interior:
        ExciseWithBoundaryCondition:
          ApparentHorizon:
            Center: [*x_left, 0., 0.]
            # $\Omega_r = \chi/(2\mathring{r}_{\min})$, *Eq.* (4.14)
            Rotation: [0., -0.25257731958762886, 0.38917525773195877]
            Lapse: *kerr_left
            NegativeExpansion: *kerr_left
      UseLogarithmicMap: True
    ObjectB:
      InnerRadius: 0.45
      OuterRadius: 4.
```

```yaml
      XCoord: *x_right
      Shape: None
      Interior:
        ExciseWithBoundaryCondition:
          ApparentHorizon:
            Center: [*x_right, 0., 0.]
            Rotation: [0., 0., 0.]
            Lapse: *kerr_right
            NegativeExpansion: *kerr_right
      UseLogarithmicMap: True
    UseEquiangularMap: True
    EnvelopingCube:
      Radius: 60.
      UseProjectiveMap: True
      Sphericity: 1.
    OuterShell:
      InnerRadius: Auto
      OuterRadius: 300.
      RadialDistribution: Inverse
      BoundaryCondition: Flatness
    InitialRefinement:
      ObjectAShell:              [{{L+1}}, {{L+1}}, {{L+1}}]
      ObjectBShell:              [{{L+1}}, {{L+1}}, {{L+1}}]
      ObjectACube:               [{{L+1}}, {{L+1}}, {{L+1}}]
      ObjectBCube:               [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeUpperZLeft:  [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeLowerZLeft:  [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeUpperYLeft:  [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeLowerYLeft:  [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeLowerX:      [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeUpperZRight: [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeLowerZRight: [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeUpperYRight: [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeLowerYRight: [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeUpperX:      [{{L+1}}, {{L+1}}, {{L+1}}]
      OuterShellUpperZLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerZLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellUpperYLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerYLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerX:          [{{L+1}}, {{L+1}}, {{L  }}]
      OuterShellUpperZRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerZRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellUpperYRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerYRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellUpperX:          [{{L+1}}, {{L+1}}, {{L  }}]
    InitialGridPoints:
      ObjectAShell:   [{{P+1}}, {{P+1}}, {{P+5}}]
      ObjectBShell:   [{{P+1}}, {{P+1}}, {{P+5}}]
      ObjectACube:    [{{P+1}}, {{P+1}}, {{P+2}}]
      ObjectBCube:    [{{P+1}}, {{P+1}}, {{P+2}}]
      EnvelopingCube: [{{P+1}}, {{P+1}}, {{P+1}}]
      OuterShell:     [{{P+1}}, {{P+1}}, {{P+1}}]

Discretization:
  DiscontinuousGalerkin:
    PenaltyParameter: 1.
    Massive: True

Observers:
  VolumeFileName: "BbhVolume"
  ReductionFileName: "BbhReductions"

NonlinearSolver:
  NewtonRaphson:
    ConvergenceCriteria:
      MaxIterations: 20
      RelativeResidual: 0.
      AbsoluteResidual: 1.e-10
    SufficientDecrease: 1.e-4
    MaxGlobalizationSteps: 40
    Damping: None
    Verbosity: Verbose

LinearSolver:
  Gmres:
    ConvergenceCriteria:
      MaxIterations: 100
      RelativeResidual: 1.e-3
      AbsoluteResidual: 1.e-12
    Verbosity: Quiet

  Multigrid:
    Iterations: 1
    MaxLevels: Auto
    PreSmoothing: True
    PostSmoothingAtBottom: True
    Verbosity: Silent
    OutputVolumeData: False
    ElementAllocation:
      WeightByNumPoints: True

  SchwarzSmoother:
    MaxOverlap: 2
    Iterations: 3
    Verbosity: Silent
    SubdomainSolver:
      Gmres:
        ConvergenceCriteria:
          MaxIterations: 3
          RelativeResidual: 1.e-4
          AbsoluteResidual: 1.e-10
        Verbosity: Silent
        Restart: None
        Preconditioner:
          MinusLaplacian:
            Solver:
              ExplicitInverse:
                FillFactor: 1
                Verbosity: Silent
            BoundaryConditions: Auto
    SkipResets: True
    ObservePerCoreReductions: False

EventsAndTriggers:
  ? HasConverged
  : - ObserveAtPoint:
        Coordinates: [0, 0, 0]
        SubfileName: Origin
        TensorsToObserve:
          - ConformalFactor
          - LapseTimesConformalFactor
          - ShiftExcess
    - ObserveAtPoint:
        Coordinates: [-5.14, 0, 0]
        SubfileName: NearLeftHorizon
        TensorsToObserve:
          - ConformalFactor
          - LapseTimesConformalFactor
          - ShiftExcess
    - ObserveAtPoint:
        Coordinates: [12.11, 0, 0]
        SubfileName: NearRightHorizon
        TensorsToObserve:
          - ConformalFactor
          - LapseTimesConformalFactor
          - ShiftExcess
    - ObserveAtPoint:
        Coordinates: [100, 0, 0]
        SubfileName: FarField
        TensorsToObserve:
          - ConformalFactor
          - LapseTimesConformalFactor
          - ShiftExcess
    - ObserveNorms:
        SubfileName: Norms
        TensorsToObserve:
          - Name: ConformalFactor
            NormType: Max
            Components: Individual
          - Name: Lapse
            NormType: Min
            Components: Individual
          - Name: Magnitude(ShiftExcess)
            NormType: Max
            Components: Individual
          - Name: HamiltonianConstraint
            NormType: L2Norm
            Components: Individual
          - Name: MomentumConstraint
            NormType: L2Norm
```

```yaml
            Components: Individual
    - ObserveFields:
        SubfileName: VolumeData
        VariablesToObserve:
          - ConformalFactor
          - Lapse
          - Shift
          - ShiftExcess
          - Magnitude(ShiftExcess)
          - SpatialMetric
          - ExtrinsicCurvature
          - HamiltonianConstraint
          - MomentumConstraint
        InterpolateToMesh: None
        CoordinatesFloatingPointType: Double
        FloatingPointTypes: [Double]
```

B.7 KerrSchildConstraints.yaml

```yaml
Background: &kerr
  KerrSchild:
    Mass: 1.
    Spin: [0., 0., 0.]
    Center: [0., 0., 0.]

InitialGuess: *kerr

DomainCreator:
  Shell:
    InnerRadius: 2.
    OuterRadius: 10.
    RadialPartitioning: []
    RadialDistribution: [Logarithmic]
    InitialRefinement: 1
    InitialGridPoints: [6, 6]
    UseEquiangularMap: True
    EquatorialCompression: None
    WhichWedges: All
    TimeDependence: None
    Shape: None
    BoundaryConditions:
      InnerBoundary:
        ApparentHorizon:
          Center: [0., 0., 0.]
          Rotation: [0., 0., 0.]
          Lapse: *kerr
          NegativeExpansion: None
      OuterBoundary:
        AnalyticSolution:
          ConformalFactor: Dirichlet
          LapseTimesConformalFactor: Dirichlet
          ShiftExcess: Dirichlet

Discretization:
  DiscontinuousGalerkin:
    PenaltyParameter: 1.
    Massive: True

Observers:
  VolumeFileName: "KerrSchildVolume"
  ReductionFileName: "KerrSchildReductions"

NonlinearSolver:
  NewtonRaphson:
    ConvergenceCriteria:
      MaxIterations: 20
      RelativeResidual: 0.
      AbsoluteResidual: 1.e-10
    SufficientDecrease: 1.e-4
    MaxGlobalizationSteps: 40
    Damping: None
    Verbosity: Verbose

LinearSolver:
  Gmres:
    ConvergenceCriteria:
      MaxIterations: 100
      RelativeResidual: 1.e-4
      AbsoluteResidual: 1.e-11
    Verbosity: Quiet

  Multigrid:
    Iterations: 1
    MaxLevels: Auto
    PreSmoothing: True
    PostSmoothingAtBottom: False
    Verbosity: Silent
    OutputVolumeData: False
    ElementAllocation:
      WeightByNumPoints: True

  SchwarzSmoother:
    MaxOverlap: 2
    Iterations: 3
    Verbosity: Silent
    SubdomainSolver:
      Gmres:
        ConvergenceCriteria:
          MaxIterations: 3
          RelativeResidual: 1.e-4
          AbsoluteResidual: 1.e-10
        Verbosity: Silent
        Restart: None
        Preconditioner:
          MinusLaplacian:
            Solver:
              ExplicitInverse:
                FillFactor: 1
                Verbosity: Silent
            BoundaryConditions: Auto
    ObservePerCoreReductions: False
    SkipResets: True

EventsAndTriggers:
  ? Always
  : - ObserveNorms:
        SubfileName: ErrorNorms
        TensorsToObserve:
          - Name: Error(ConformalFactor)
            NormType: L2Norm
            Components: Individual
          - Name: Error(LapseTimesConformalFactor)
            NormType: L2Norm
            Components: Individual
          - Name: Error(ShiftExcess)
            NormType: L2Norm
            Components: Individual
    - ObserveNorms:
        SubfileName: Norms
        TensorsToObserve:
          - Name: HamiltonianConstraint
            NormType: L2Norm
            Components: Individual
          - Name: MomentumConstraint
            NormType: L2Norm
            Components: Individual
    - ObserveFields:
        SubfileName: VolumeData
        VariablesToObserve:
          - ConformalFactor
          - Lapse
          - Shift
          - Error(ConformalFactor)
          - Error(LapseTimesConformalFactor)
          - Error(ShiftExcess)
          - HamiltonianConstraint
          - MomentumConstraint
```

```yaml
        InterpolateToMesh: None
        CoordinatesFloatingPointType: Float
        FloatingPointTypes: [Float]
```

B.8 Tov.yaml

```yaml
Background: &background
  TovStar:
    CentralDensity: 0.0008087415253997405  # h = 1.2
    PolytropicConstant: 123.6489
    PolytropicExponent: 2.
    Coordinates: Isotropic

InitialGuess: *background

DomainCreator:
  Sphere:
    InnerRadius: 3.
    OuterRadius: 18.
    # For configuration "S":
    RadialPartitioning: [&star_radius 9.709353324763269]
    InitialRefinement: 1
    # For configurations "L1" to "L3":
    # RadialPartitioning: [6.]
    # InitialRefinement:
    #   InnerCube: [1, 1,  L ]
    #   Shell0: [1, 1,  L - 1 ]
    #   Shell1: [1, 1,  L + 1 ]
    InitialGridPoints: [{{P+1}}, {{P+1}}, {{P+1}}]
    RadialDistribution: [Linear, Linear]
    UseEquiangularMap: True
    BoundaryCondition:
      AnalyticSolution:
        ConformalFactor: Dirichlet
        LapseTimesConformalFactor: Dirichlet
        ShiftExcess: Dirichlet

Discretization:
  DiscontinuousGalerkin:
    PenaltyParameter: 1.
    Massive: True

Observers:
  VolumeFileName: "TovVolume"
  ReductionFileName: "TovReductions"

NonlinearSolver:
  NewtonRaphson:
    ConvergenceCriteria:
      MaxIterations: 20
      RelativeResidual: 0.
      AbsoluteResidual: 1.e-10
    SufficientDecrease: 1.e-4
    MaxGlobalizationSteps: 40
    Damping: None
    Verbosity: Verbose

LinearSolver:
  Gmres:
    ConvergenceCriteria:
      MaxIterations: 100
      RelativeResidual: 1.e-3
      AbsoluteResidual: 1.e-10
    Verbosity: Quiet

  Multigrid:
    Iterations: 1
    MaxLevels: Auto
    PreSmoothing: True
    PostSmoothingAtBottom: False
    Verbosity: Silent
    OutputVolumeData: False
    ElementAllocation:
      WeightByNumPoints: True

  SchwarzSmoother:
    MaxOverlap: 2
    Iterations: 3
    Verbosity: Silent
    SubdomainSolver:
      Gmres:
        ConvergenceCriteria:
          MaxIterations: 3
          RelativeResidual: 1.e-4
          AbsoluteResidual: 1.e-10
        Verbosity: Silent
        Restart: None
        Preconditioner:
          MinusLaplacian:
            Solver:
              ExplicitInverse:
                FillFactor: 1
                Verbosity: Silent
            BoundaryConditions: Auto
    SkipResets: True
    ObservePerCoreReductions: False

EventsAndTriggers:
  ? HasConverged
  : - ObserveNorms:
        SubfileName: ErrorNorms
        TensorsToObserve:
          - Name: Error(ConformalFactor)
            NormType: L2Norm
            Components: Individual
          - Name: Error(ConformalFactor)
            NormType: L2IntegralNorm
            Components: Individual
          - Name: Error(LapseTimesConformalFactor)
            NormType: L2Norm
            Components: Individual
          - Name: Error(LapseTimesConformalFactor)
            NormType: L2IntegralNorm
            Components: Individual
          - Name: Error(ShiftExcess)
            NormType: L2Norm
            Components: Individual
          - Name: Error(ShiftExcess)
            NormType: L2IntegralNorm
            Components: Individual
    - ObserveNorms:
        SubfileName: Norms
        TensorsToObserve:
          - Name: ConformalFactor
            NormType: Max
            Components: Individual
          - Name: Lapse
            NormType: Min
            Components: Individual
          - Name: HamiltonianConstraint
            NormType: L2Norm
            Components: Individual
          - Name: MomentumConstraint
            NormType: L2Norm
            Components: Individual
    - ObserveFields:
        SubfileName: VolumeData
        VariablesToObserve:
          - ConformalFactor
          - Lapse
          - Conformal(EnergyDensity)
          - Conformal(StressTrace)
          - Error(ConformalFactor)
          - Error(LapseTimesConformalFactor)
          - Error(ShiftExcess)
          - HamiltonianConstraint
```

```
          - MomentumConstraint
        InterpolateToMesh: None
        CoordinatesFloatingPointType: Float
        FloatingPointTypes: [Float]
```

B.9 BnsHeadOn.yaml

```yaml
Background: &binary
  Binary:
    XCoords: [&x_left -20., &x_right 20.]
    ObjectA: &star
      TovStar:
        CentralDensity: 0.0008087415253997405  # h = 1.2
        PolytropicConstant: 123.6489
        PolytropicExponent: 2.
        Coordinates: Isotropic
    ObjectB: *star
    AngularVelocity: 0.
    Expansion: 0.
    FalloffWidths: None

InitialGuess: *binary

DomainCreator:
  BinaryCompactObject:
    ObjectA:
      InnerRadius: 4.
      OuterRadius: &star_radius 9.709353324763269
      XCoord: *x_left
      Interior: Auto
      UseLogarithmicMap: False
      Shape: None
    ObjectB:
      InnerRadius: 4.
      OuterRadius: *star_radius
      XCoord: *x_right
      Interior: Auto
      UseLogarithmicMap: False
      Shape: None
    UseEquiangularMap: True
    EnvelopingCube:
      Radius: 120.
      UseProjectiveMap: True
      Sphericity: 1.
    OuterShell:
      InnerRadius: Auto
      OuterRadius: 600.
      RadialDistribution: Inverse
      BoundaryCondition: Flatness
    InitialRefinement:
      ObjectAInterior:           [{{L+1}}, {{L+1}}, {{L+1}}]
      ObjectBInterior:           [{{L+1}}, {{L+1}}, {{L+1}}]
      ObjectAShell:              [{{L+1}}, {{L+1}}, {{L+1}}]
      ObjectBShell:              [{{L+1}}, {{L+1}}, {{L+1}}]
      ObjectACube:               [{{L+1}}, {{L+1}}, {{L+1}}]
      ObjectBCube:               [{{L+1}}, {{L+1}}, {{L+1}}]
      EnvelopingCubeUpperZLeft:  [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeLowerZLeft:  [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeUpperYLeft:  [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeLowerYLeft:  [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeLowerX:      [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeUpperZRight: [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeLowerZRight: [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeUpperYRight: [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeLowerYRight: [{{L+1}}, {{L+1}}, {{L  }}]
      EnvelopingCubeUpperX:      [{{L+1}}, {{L+1}}, {{L  }}]
      OuterShellUpperZLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerZLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellUpperYLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerYLeft:      [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerX:          [{{L+1}}, {{L+1}}, {{L  }}]
      OuterShellUpperZRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerZRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellUpperYRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellLowerYRight:     [{{L  }}, {{L+1}}, {{L  }}]
      OuterShellUpperX:          [{{L+1}}, {{L+1}}, {{L  }}]
      ObjectAInterior: [{{P+1}}, {{P+1}}, {{P+1}}]
      ObjectBInterior: [{{P+1}}, {{P+1}}, {{P+1}}]
      ObjectAShell:    [{{P+1}}, {{P+1}}, {{P+2}}]
      ObjectBShell:    [{{P+1}}, {{P+1}}, {{P+2}}]
      ObjectACube:     [{{P+1}}, {{P+1}}, {{P+2}}]
      ObjectBCube:     [{{P+1}}, {{P+1}}, {{P+2}}]
      EnvelopingCube:  [{{P+1}}, {{P+1}}, {{P+1}}]
      OuterShell:      [{{P+1}}, {{P+1}}, {{P  }}]

Discretization:
  DiscontinuousGalerkin:
    PenaltyParameter: 1.
    Massive: True

Observers:
  VolumeFileName: "BnsVolume"
  ReductionFileName: "BnsReductions"

NonlinearSolver:
  NewtonRaphson:
    ConvergenceCriteria:
      MaxIterations: 20
      RelativeResidual: 0.
      AbsoluteResidual: 1.e-10
    SufficientDecrease: 1.e-4
    MaxGlobalizationSteps: 40
    Damping: None
    Verbosity: Verbose

LinearSolver:
  Gmres:
    ConvergenceCriteria:
      MaxIterations: 100
      RelativeResidual: 1.e-3
      AbsoluteResidual: 1.e-10
    Verbosity: Quiet

  Multigrid:
    Iterations: 1
    MaxLevels: Auto
    PreSmoothing: True
    PostSmoothingAtBottom: True
    Verbosity: Silent
    OutputVolumeData: False
    ElementAllocation:
      WeightByNumPoints: True

  SchwarzSmoother:
    MaxOverlap: 2
    Iterations: 3
    Verbosity: Silent
    SubdomainSolver:
      Gmres:
        ConvergenceCriteria:
          MaxIterations: 3
          RelativeResidual: 1.e-4
          AbsoluteResidual: 1.e-10
        Verbosity: Silent
        Restart: None
        Preconditioner:
          MinusLaplacian:
            Solver:
              ExplicitInverse:
                FillFactor: 1
                Verbosity: Silent
            BoundaryConditions: Auto
    SkipResets: True
    ObservePerCoreReductions: False

EventsAndTriggers:
```

```
? HasConverged
: - ObserveAtPoint:
      Coordinates: [0, 0, 0]
      SubfileName: Origin
      TensorsToObserve:
        - ConformalFactor
        - LapseTimesConformalFactor
        - ShiftExcess
  - ObserveAtPoint:
      Coordinates: [-20, 0, 0]
      SubfileName: StarCenter
      TensorsToObserve:
        - ConformalFactor
        - LapseTimesConformalFactor
        - ShiftExcess
        - Conformal(EnergyDensity)
        - Conformal(StressTrace)
  - ObserveNorms:
      SubfileName: Norms
      TensorsToObserve:
        - Name: ConformalFactor
          NormType: Max
          Components: Individual
        - Name: Lapse
          NormType: Min
          Components: Individual
        - Name: Magnitude(ShiftExcess)
          NormType: Max
          Components: Individual
        - Name: Conformal(EnergyDensity)
          NormType: Max
          Components: Individual
        - Name: Conformal(StressTrace)
          NormType: Max
          Components: Individual
        - Name: HamiltonianConstraint
          NormType: L2Norm
          Components: Individual
        - Name: MomentumConstraint
          NormType: L2Norm
          Components: Individual
  - ObserveFields:
      SubfileName: VolumeData
      VariablesToObserve:
        - ConformalFactor
        - Lapse
        - Shift
        - HamiltonianConstraint
        - MomentumConstraint
        - Conformal(EnergyDensity)
        - Conformal(StressTrace)
      InterpolateToMesh: None
      CoordinatesFloatingPointType: Float
      FloatingPointTypes: [Float]
```

References

[1] **N. L. Fischer** and H. P. Pfeiffer, "Unified discontinuous Galerkin scheme for a large class of elliptic equations," Phys. Rev. D **105**, 024034 (2022), arXiv:2108.05826, Chapter 2 of this thesis. (Cit. on pp. xiii, 1, 32, 37, 40, 71–73, 75, 76, 81, 82, 85, 90, 91, 95, 96, 102, 107, 136, 137, 139, 141, 142, 151).

[2] **N. L. Vu** et al., "A scalable elliptic solver with task-based parallelism for the SpECTRE code," Phys. Rev. D **105**, 084027 (2022), arXiv:2111.06767, Chapter 3 of this thesis. (Cit. on pp. xiii, 1, 34, 38, 58, 66, 67, 69, 107, 116, 119, 137, 143, 144, 148, 151).

[3] **N. L. Vu** et al., "High-accuracy numerical models of Brownian thermal noise in thin mirror coatings," Submitted to Phys. Rev. D, Nov. 2021, arXiv:2111.06893, Chapter 5 of this thesis. (Cit. on pp. xiii, 1, 2, 104, 135, 152).

[4] S. Ma et al., "Gravitational-wave echoes from numerical-relativity waveforms via spacetime construction near merging compact objects," Phys. Rev. D **105**, 104007 (2022), arXiv:2203.03174 (cit. on p. xiii).

[5] L. M. Zertuche et al., "High precision ringdown modeling: multimode fits and BMS frames," Phys. Rev. D **105**, 104015 (2022), arXiv:2110.15922 (cit. on p. xiii).

[6] J. Moxon et al., "The SpECTRE Cauchy-characteristic evolution system for rapid, precise waveform extraction," Submitted to Phys. Rev. D, Oct. 2021, arXiv:2110.08635 (cit. on pp. xiii, 12).

[7] N. Deppe et al., "Simulating magnetized neutron stars with discontinuous Galerkin methods," Submitted to Phys. Rev. D, Sept. 2021, arXiv:2109.12033 (cit. on pp. xiii, 13, 107).

[8] T. Vincent, H. P. Pfeiffer, and **N. L. Fischer**, "hp-adaptive discontinuous Galerkin solver for elliptic equations in numerical relativity," Phys. Rev. D **100**, 084052 (2019), arXiv:1907.01572 (cit. on pp. xiii, 38, 41, 48, 50, 53, 71, 81, 82, 84, 85, 87, 131).

[9] M. Boyle et al. (SXS), "The SXS collaboration catalog of binary black hole simulations," Classical and Quantum Gravity **36**, 195006 (2019), arXiv:1904.04831 (cit. on pp. xiii, 11, 13, 118, 157).

[10] N. Deppe et al., *SpECTRE*, spectre-code.org, version 2022.04.04, Apr. 2022 (cit. on pp. xiii, 1, 12, 36, 38, 58, 66, 67, 71, 79, 104, 106, 137, 143, 149, 151, 161).

[11] **N. L. Vu**, *dgpy*, 10.5281/zenodo.5086181, version 0.1, July 2021 (cit. on pp. xiii, 67, 106, 149).

[12] **N. L. Vu**, *gwpv*, github:nilsvu/gwpv (cit. on pp. xiii, 157).

[13] M. Düll et al., "Symmetric gravitational closure," 2020, arXiv:2003.07109 (cit. on pp. xiii, 15).

[14] V. Kuznetsov, **N. L. Fischer**, and Y. Guo, "The archive solution for distributed workflow management agents of the CMS experiment at LHC," Computing and Software for Big Science **2**, 10.1007/s41781-018-0005-0 (2018), arXiv:1801.03872 (cit. on p. xiii).

[15] S. M. Carroll, *Spacetime and geometry, An introduction to general relativity* (Addison Wesley, 2004) (cit. on pp. 2, 3, 6, 110, 112).

[16] K. S. Thorne and R. D. Blandford, *Modern classical physics, Optics, fluids, plasmas, elasticity, relativity, and statistical physics* (Princeton University Press, Sept. 2017) (cit. on pp. 2, 4, 59, 138).

[17] T. W. Baumgarte and S. L. Shapiro, *Numerical relativity: solving Einsteins equations on the computer* (Cambridge University Press, Cambridge, England, June 2010) (cit. on pp. 3, 5, 16–25, 61, 64, 70, 95, 105, 115, 117, 126, 127, 132).

[18] B. P. Abbott et al. (LIGO, Virgo), "Observation of gravitational waves from a binary black hole merger," Phys. Rev. Lett. **116**, 061102 (2016), arXiv:1602.03837 (cit. on pp. 4, 6, 7, 70).

[19] L. Blanchet, "Gravitational radiation from post-Newtonian sources and inspiralling compact binaries," Living Rev. Rel. **17**, 2 (2014), arXiv:1310.1528 (cit. on p. 5).

[20] F. Pretorius, "Evolution of binary black hole spacetimes," Phys. Rev. Lett. **95**, 121101 (2005), arXiv:gr-qc/0507014 (cit. on pp. 5, 19).

[21] M. Campanelli et al., "Accurate evolutions of orbiting black-hole binaries without excision," Phys. Rev. Lett. **96**, 111101 (2006), arXiv:gr-qc/0511048 (cit. on pp. 5, 19).

[22] J. G. Baker et al., "Gravitational wave extraction from an inspiraling configuration of merging black holes," Phys. Rev. Lett. **96**, 111102 (2006), arXiv:gr-qc/0511103 (cit. on pp. 5, 19).

[23] S. A. Teukolsky, "Perturbations of a rotating black hole. 1. fundamental equations for gravitational electromagnetic and neutrino field perturbations," Astrophys. J. **185**, 635 (1973) (cit. on p. 6).

[24] A. Buonanno, G. B. Cook, and F. Pretorius, "Inspiral, merger and ring-down of equal-mass black-hole binaries," Phys. Rev. D **75**, 124018 (2007), arXiv:gr-qc/0610122 (cit. on p. 6).

[25] E. Berti, V. Cardoso, and A. O. Starinets, "Quasinormal modes of black holes and black branes," Class. Quant. Grav. **26**, 163001 (2009), arXiv:0905.2975 (cit. on p. 6).

[26] E. Berti and A. Klein, "Mixing of spherical and spheroidal modes in perturbed Kerr black holes," Phys. Rev. D **90**, 064012 (2014), arXiv:1408.1860 (cit. on p. 6).

[27] J. Aasi et al. (LIGO), "Advanced LIGO," Class. Quant. Grav. **32**, 074001 (2015), arXiv:1411.4547 (cit. on pp. 6, 7).

[28] B. P. Abbott et al. (LIGO, Virgo), "A guide to LIGO–Virgo detector noise and extraction of transient gravitational-wave signals," Class. Quant. Grav. **37**, 055002 (2020), arXiv:1908.11170 (cit. on pp. 6–8, 10, 11).

[29] L. Barsotti et al., *The a+ design curve*, tech. rep. T1800042-v5 (LIGO Document, 2018) (cit. on pp. 7, 136).

[30] T. Akutsu et al. (KAGRA), "The status of KAGRA underground cryogenic gravitational wave telescope," Journal of Physics: Conference Series **1342**, 012014 (2020) (cit. on p. 7).

[31] B. P. Abbott et al. (LIGO, Virgo), "Properties of the binary black hole merger GW150914," Phys. Rev. Lett. **116**, 241102 (2016), arXiv:1602.03840 (cit. on p. 9).

[32] B. P. Abbott et al. (LIGO, Virgo), "GWTC-1: a gravitational-wave transient catalog of compact binary mergers observed by LIGO and Virgo during the first and second observing runs," Phys. Rev. X **9**, 031040 (2019), arXiv:1811.12907 (cit. on pp. 9, 70).

[33] R. Abbott et al. (LIGO, Virgo), "GWTC-2: compact binary coalescences observed by LIGO and Virgo during the first half of the third observing run," Phys. Rev. X **11**, 021053 (2021), arXiv:2010.14527 (cit. on pp. 9, 70).

[34] R. Abbott et al. (LIGO, Virgo, KAGRA), "GWTC-3: compact binary coalescences observed by LIGO and Virgo during the second part of the third observing run," Nov. 2021, arXiv:2111.03606 (cit. on pp. 9, 70).

[35] R. Abbott et al. (LIGO, Virgo), "GW190412: observation of a binary-black-hole coalescence with asymmetric masses," Phys. Rev. D **102**, 043015 (2020), arXiv:2004.08342 (cit. on pp. 10, 157).

[36] R. Abbott et al. (LIGO, Virgo), "GW190814: gravitational waves from the coalescence of a 23 solar mass black hole with a 2.6 solar mass compact object," Astrophys. J. Lett. **896**, L44 (2020), arXiv:2006.12611 (cit. on pp. 10, 157).

[37] R. Abbott et al. (LIGO, Virgo), "GW190521: a binary black hole merger with a total mass of $150M_{\odot}$," Phys. Rev. Lett. **125**, 101102 (2020), arXiv:2009.01075 (cit. on pp. 10, 157).

[38] A. Buonanno and T. Damour, "Effective one-body approach to general relativistic two-body dynamics," Phys. Rev. D **59**, 084006 (1999), arXiv:gr-qc/9811091 (cit. on p. 10).

[39] A. Bohé et al., "Improved effective-one-body model of spinning, nonprecessing binary black holes for the era of gravitational-wave astrophysics with advanced detectors," Phys. Rev. D **95**, 044028 (2017), arXiv:1611.03703 (cit. on pp. 10, 11).

[40] S. Ossokine et al., "Multipolar effective-one-body waveforms for precessing binary black holes: construction and validation," Phys. Rev. D **102**, 044055 (2020), arXiv:2004.09442 (cit. on p. 10).

[41] A. Antonelli et al., "Energetics of two-body Hamiltonians in post-minkowskian gravity," Phys. Rev. D **99**, 104004 (2019), arXiv:1901.07102 (cit. on p. 11).

[42] A. Antonelli et al., "Quasicircular inspirals and plunges from nonspinning effective-one-body Hamiltonians with gravitational self-force information," Phys. Rev. D **101**, 024024 (2020), arXiv:1907.11597 (cit. on p. 11).

[43] E. Barausse and A. Buonanno, "An improved effective-one-body Hamiltonian for spinning black-hole binaries," Phys. Rev. D **81**, 084024 (2010), arXiv:0912.3517 (cit. on p. 11).

[44] A. Taracchini et al., "Effective-one-body model for black-hole binaries with generic mass ratios and spins," Phys. Rev. D **89**, 061502 (2014), arXiv:1311.2544 (cit. on p. 11).

[45] T. Damour and A. Nagar, "A new analytic representation of the ringdown waveform of coalescing spinning black hole binaries," Phys. Rev. D **90**, 024054 (2014), arXiv:1406.0401 (cit. on p. 11).

[46] R. Cotesta et al., "Enriching the symphony of gravitational waves from binary black holes by tuning higher harmonics," Phys. Rev. D **98**, 084028 (2018), arXiv:1803.10701 (cit. on pp. 11, 157).

[47] S. Khan et al., "Frequency-domain gravitational waves from nonprecessing black-hole binaries. II. a phenomenological model for the advanced detector era," Phys. Rev. D **93**, 044007 (2016), arXiv:1508.07253 (cit. on p. 11).

[48] J. Blackman et al., "Numerical relativity waveform surrogate model for generically precessing binary black hole mergers," Phys. Rev. D **96**, 024058 (2017), arXiv:1705.07089 (cit. on p. 11).

[49] V. Varma et al., "Surrogate model of hybridized numerical relativity binary black hole waveforms," Phys. Rev. D **99**, 064045 (2019), arXiv:1812.07865 (cit. on p. 11).

[50] J. Yoo et al., "Targeted large mass ratio numerical relativity surrogate waveform model for GW190814," (2022), arXiv:2203.10109 (cit. on p. 11).

[51] V. Varma et al., "Surrogate models for precessing binary black hole simulations with unequal masses," Phys. Rev. Research. **1**, 033015 (2019), arXiv:1905.09300 (cit. on p. 11).

[52] T. Dietrich et al., "CoRe database of binary neutron star merger waveforms," Class. Quant. Grav. **35**, 24LT01 (2018), arXiv:1806.01625 (cit. on p. 11).

[53] J. Healy and C. O. Lousto, "The fourth RIT binary black hole simulations catalog: extension to eccentric orbits," (2022), arXiv:2202.00018 (cit. on p. 11).

[54] K. Jani et al., "Georgia Tech catalog of gravitational waveforms," Class. Quant. Grav. **33**, 204001 (2016), arXiv:1605.03204 (cit. on p. 11).

[55] L. E. Kidder, H. P. Pfeiffer, M. A. Scheel, et al., *Spectral Einstein Code (SpEC)*, black-holes.org/code/SpEC (cit. on pp. 11, 36, 66, 70, 102, 103, 108, 151).

[56] P. Mészáros et al., "Multi-messenger astrophysics," Nature Rev. Phys. **1**, 585 (2019), arXiv:1906.10212 (cit. on pp. 11, 12).

[57] National Academies of Sciences, Engineering, and Medicine, *Pathways to discovery in astronomy and astrophysics for the 2020s (decadal survey)* (The National Academies Press, Washington, DC, 2021) (cit. on p. 11).

[58] R. J. Foley et al., "Gravity and light: combining gravitational wave and electromagnetic observations in the 2020s," (2019), arXiv:1903.04553 (cit. on p. 11).

[59] V. Kalogera et al., "The next generation global gravitational wave observatory: the science book," (2021), arXiv:2111.06990 (cit. on pp. 11, 12).

[60] B. P. Abbott et al. (LIGO, Virgo), "GW170817: observation of gravitational waves from a binary neutron star inspiral," Phys. Rev. Lett. **119**, 161101 (2017), arXiv:1710.05832 (cit. on pp. 11, 70).

[61] B. P. Abbott et al. (LIGO Scientific, Virgo, Fermi-GBM, INTEGRAL), "Gravitational waves and gamma-rays from a binary neutron star merger: GW170817 and GRB 170817A," Astrophys. J. Lett. **848**, L13 (2017), arXiv:1710.05834 (cit. on p. 11).

[62] B. P. Abbott et al. (LIGO, Virgo, et al.), "Multi-messenger observations of a binary neutron star merger," Astrophys. J. Lett. **848**, L12 (2017), arXiv:1710.05833 (cit. on p. 11).

[63] T. Dietrich, T. Hinderer, and A. Samajdar, "Interpreting binary neutron star mergers, Describing the binary neutron star dynamics, modelling gravitational waveforms, and analyzing detections," Gen. Rel. Grav. **53**, 27 (2021), arXiv:2004.02527 (cit. on pp. 12, 13).

[64] M. Evans et al., "A horizon study for Cosmic Explorer: science, observatories, and community," (2021), arXiv:2109.09882 (cit. on p. 12).

[65] M. Maggiore et al., "Science case for the Einstein Telescope," JCAP **03**, 050 (2020), arXiv:1912.02622 (cit. on p. 12).

[66] P. A. Seoane et al. (eLISA), "The gravitational universe," (2013), arXiv:1305.5720 (cit. on p. 12).

[67] M. Pürrer and C.-J. Haster, "Gravitational waveform accuracy requirements for future ground-based detectors," Phys. Rev. Res. **2**, 023151 (2020), arXiv:1912.10055 (cit. on p. 12).

[68] D. Ferguson et al., "Assessing the readiness of numerical relativity for LISA and 3G detectors," Phys. Rev. D **104**, 044037 (2021), arXiv:2006.04272 (cit. on p. 12).

[69] E. R. Most, L. J. Papenfort, and L. Rezzolla, "Beyond second-order convergence in simulations of magnetized binary neutron stars with realistic microphysics," Mon. Not. Roy. Astron. Soc. **490**, 3588 (2019), arXiv:1907.10328 (cit. on p. 13).

[70] F. Foucart et al., "Monte-Carlo neutrino transport in neutron star merger simulations," Astrophys. J. Lett. **902**, L27 (2020), arXiv:2008.08089 (cit. on p. 13).

[71] S. Brandt and B. Brügmann, "A simple construction of initial data for multiple black holes," Phys. Rev. Lett. **78**, 3606 (1997), arXiv:gr-qc/9703066 (cit. on pp. 13, 18).

[72] S. Dain, C. O. Lousto, and Y. Zlochower, "Extra-large remnant recoil velocities and spins from near-extremal-Bowen-York-spin black-hole binaries," Phys. Rev. D **78**, 024039 (2008), arXiv:0803.0351 (cit. on p. 13).

[73] S. Dain, C. O. Lousto, and R. Takahashi, "New conformally flat initial data for spinning black holes," Phys. Rev. D **65**, 104038 (2002), arXiv:gr-qc/0201062 (cit. on p. 13).

[74] H. P. Pfeiffer, "The initial value problem in numerical relativity," J. Hyperbolic Differ. Equ. **2**, 497 (2005), arXiv:gr-qc/0412002 (cit. on pp. 13, 21, 38, 64, 70, 73).

[75] G. Lovelace et al., "Binary-black-hole initial data with nearly-extremal spins," Phys. Rev. D **78**, 084017 (2008), arXiv:0805.4192 (cit. on pp. 13, 21, 38, 64, 100, 108, 111, 115, 116, 118).

[76] I. Ruchlin et al., "Puncture initial data for black-hole binaries with high spins and high boosts," Phys. Rev. D **95**, 024033 (2017), arXiv:1410.8607 (cit. on pp. 13, 20).

[77] G. Lovelace, "Reducing spurious gravitational radiation in binary-black-hole simulations by using conformally curved initial data," Class. Quant. Grav. **26**, edited by P. Sutton and D. Shoemaker, 114002 (2009), arXiv:0812.3132 (cit. on p. 13).

[78] V. Varma, M. A. Scheel, and H. P. Pfeiffer, "Comparison of binary black hole initial data sets," Phys. Rev. D **98**, 104011 (2018), arXiv:1808.08228 (cit. on pp. 14, 21, 38, 64, 65, 100, 102, 104, 108, 113, 117, 118, 124).

[79] A. Tsokaros, K. Uryū, and S. L. Shapiro, "Complete initial value spacetimes containing black holes in general relativity: application to black hole-disk systems," Phys. Rev. D **99**, 041501 (2019), arXiv:1810.02825 (cit. on p. 14).

[80] P. Grandclément and J. Nicoules, "Boundary conditions for stationary black holes: application to Kerr, Martínez-Troncoso-Zanelli, and hairy black holes," Phys. Rev. D **105**, 104011 (2022), arXiv:2203.09341 (cit. on p. 14).

[81] K. Uryū et al., "New code for equilibriums and quasiequilibrium initial data of compact objects. III. axisymmetric and triaxial rotating stars," Phys. Rev. D **93**, 044056 (2016) (cit. on pp. 14, 134).

[82] K. Kyutoku, M. Shibata, and K. Taniguchi, "Coalescence of black hole–neutron star binaries," Living Rev. Rel. **24**, 5 (2021), arXiv:2110.06218 (cit. on p. 14).

[83] K. Kiuchi et al., "High resolution numerical-relativity simulations for the merger of binary magnetized neutron stars," Phys. Rev. D **90**, 041502 (2014), arXiv:1407.2660 (cit. on p. 14).

[84] K. Hayashi et al., "General-relativistic neutrino-radiation magnetohydrodynamics simulation of black hole-neutron star mergers for seconds," (2021), arXiv:2111.04621 (cit. on p. 14).

[85] P. C.-K. Cheong, L.-M. Lin, and T. G.-F. Li, "Gmunu: toward multigrid based Einstein field equations solver for general-relativistic hydrodynamics simulations," 10.1088/1361-6382/ab8e9c (2020), arXiv:2001.05723 (cit. on pp. 14, 24, 153).

[86] W. E. East, F. M. Ramazanoglu, and F. Pretorius, "Conformal Thin-Sandwich solver for generic initial data," Phys. Rev. D **86**, 104053 (2012), arXiv:1208.3473 (cit. on p. 14).

[87] N. Moldenhauer et al., "Initial data for binary neutron stars with adjustable eccentricity," Phys. Rev. D **90**, 084043 (2014), arXiv:1408.4136 (cit. on pp. 14, 23, 128, 134).

[88] M. Ansorg, B. Brügmann, and W. Tichy, "A single-domain spectral method for black hole puncture data," Phys. Rev. D **70**, 064011 (2004), arXiv:gr-qc/0404056 (cit. on pp. 14, 34, 70).

[89] L. J. Papenfort et al., "New public code for initial data of unequal-mass, spinning compact-object binaries," Phys. Rev. D **104**, 024057 (2021), arXiv:2103.09911 (cit. on pp. 14, 21, 33, 70, 77, 108, 130).

[90] H. Witek et al., "Black holes and binary mergers in scalar Gauss-Bonnet gravity: scalar field dynamics," Phys. Rev. D **99**, 064035 (2019), arXiv:1810.05177 (cit. on p. 14).

[91] M. Okounkova, "Numerical relativity simulation of GW150914 in Einstein dilaton Gauss-Bonnet gravity," Phys. Rev. D **102**, 084046 (2020), arXiv:2001.03571 (cit. on p. 14).

[92] A. M. Knapp, E. J. Walker, and T. W. Baumgarte, "Illustrating stability properties of numerical relativity in electrodynamics," (2002), arXiv:gr-qc/0201051 (cit. on p. 15).

[93] R. L. Arnowitt, S. Deser, and C. W. Misner, in *Gravitation: an introduction to current research*, edited by L. Witten (Wiley, New York, 1962) Chap. 7, pp. 227–265, arXiv:gr-qc/0405109 (cit. on p. 17).

[94] J. W. York Jr., "Kinematics and dynamics of general relativity," in Sources of gravitational radiation, edited by L. L. Smarr (1979), pp. 83–126 (cit. on pp. 17, 18).

[95] A. Lichnerowicz, "L'intégration des équations de la gravitation relativiste et la problème des n corps," J. Math. Pure Appl, 37 (1944) (cit. on p. 18).

[96] N. O Murchadha and J. W. York Jr., "Gravitational energy," Phys. Rev. D **10**, 2345 (1974) (cit. on p. 18).

[97] J. M. Bowen and J. W. York Jr., "Time asymmetric initial data for black holes and black hole collisions," Phys. Rev. D **21**, 2047 (1980) (cit. on p. 19).

[98] F. Pretorius, "Numerical relativity using a generalized harmonic decomposition," Class. Quant. Grav. **22**, 425 (2005), arXiv:gr-qc/0407110 (cit. on p. 19).

[99] B. Khamesra, M. Gracia-Linares, and P. Laguna, "Black hole–neutron star binary mergers: the imprint of tidal deformations and debris," Classical and Quantum Gravity **38**, 185008 (2021) (cit. on p. 20).

[100] M. Clark and P. Laguna, "Bowen-York type initial data for binaries with neutron stars," Phys. Rev. D **94**, 064058 (2016), arXiv:1606.04881 (cit. on pp. 20, 128).

[101] K. Kyutoku et al., "Reducing orbital eccentricity in initial data of black hole–neutron star binaries in the puncture framework," Physical Review D **103**, 10.1103/physrevd.103.023002 (2021) (cit. on p. 20).

[102] Z. B. Etienne et al., "Filling the holes: evolving excised binary black hole initial data with puncture techniques," Physical Review D **76**, 10.1103/physrevd.76.101503 (2007) (cit. on p. 20).

[103] S. V. Chaurasia, T. Dietrich, and S. Rosswog, "Black hole-neutron star simulations with the BAM code: first tests and simulations," Physical Review D **104**, 10.1103/physrevd.104.084010 (2021) (cit. on p. 20).

[104] J. W. York Jr., "Conformal 'thin sandwich' data for the initial-value problem of general relativity," Phys. Rev. Lett. **82**, 1350 (1999), arXiv:gr-qc/9810051 (cit. on p. 20).

[105] H. P. Pfeiffer and J. W. York Jr., "Extrinsic curvature and the Einstein constraints," Phys. Rev. D **67**, 044022 (2003), arXiv:gr-qc/0207095 (cit. on pp. 20, 21).

[106] R. A. Matzner, M. F. Huq, and D. Shoemaker, "Initial data and coordinates for multiple black hole systems," Phys. Rev. D **59**, 024015 (1999), arXiv:gr-qc/9805023 (cit. on p. 21).

[107] T. Dietrich et al., "Binary neutron stars with generic spin, eccentricity, mass ratio, and compactness - quasi-equilibrium sequences and first evolutions," Phys. Rev. D **92**, 124007 (2015), arXiv:1507.07100 (cit. on pp. 21, 70, 131).

[108] A. Rashti et al., "Elliptica: a new pseudo-spectral code for the construction of initial data," Sept. 2021, arXiv:2109.14511 (cit. on pp. 21, 33, 70, 104, 108, 130, 134).

[109] N. Tacik et al., "Binary neutron stars with arbitrary spins in numerical relativity," Phys. Rev. D **92**, 124012 (2015), arXiv:1508.06986 (cit. on pp. 21, 38, 70).

[110] A. Tsokaros, K. Uryū, and L. Rezzolla, "New code for quasiequilibrium initial data of binary neutron stars: corotating, irrotational, and slowly spinning systems," Phys. Rev. D **91**, 104030 (2015), arXiv:1502.05674 (cit. on pp. 21, 70, 128, 134).

[111] K. Uryū et al., "Non-conformally flat initial data for binary compact objects," Phys. Rev. D **80**, 124004 (2009), arXiv:0908.0579 (cit. on pp. 21, 23).

[112] N. Tacik et al., "Initial data for black hole–neutron star binaries, with rotating stars," Class. Quantum Gravity **33**, 225012 (2016), arXiv:1607.07962 (cit. on pp. 21, 38, 70, 130, 134).

[113] G. B. Cook and H. P. Pfeiffer, "Excision boundary conditions for black hole initial data," Phys. Rev. D **70**, 104016 (2004), arXiv:gr-qc/0407078 (cit. on pp. 22, 38, 62, 101, 108, 116, 117).

[114] L. Rezzolla and O. Zanotti, *Relativistic hydrodynamics* (Oxford University Press, Oxford, UK, 2013) (cit. on pp. 23, 126, 128).

[115] C. Gundlach, "Pseudospectral apparent horizon finders: an efficient new algorithm," Phys. Rev. D **57**, 863 (1998), arXiv:gr-qc/9707050 (cit. on pp. 24, 106, 115).

[116] S. Bonazzola et al., "A constrained scheme for Einstein equations based on Dirac gauge and spherical coordinates," Phys. Rev. D **70**, 104007 (2004), arXiv:gr-qc/0307082 (cit. on p. 24).

[117] C. Gundlach et al., "Constraint damping in the z4 formulation and harmonic gauge," Class. Quant. Grav. **22**, 3767 (2005), arXiv:gr-qc/0504114 (cit. on p. 24).

[118] L. Lindblom et al., "A new generalized harmonic evolution system," Class. Quant. Grav. **23**, S447 (2006), arXiv:gr-qc/0512093 (cit. on p. 24).

[119] A. Dedner et al., "Hyperbolic divergence cleaning for the mhd equations," Journal of Computational Physics **175**, 645 (2002) (cit. on p. 24).

[120] P. C.-K. Cheong et al., "An extension of Gmunu: general-relativistic resistive magnetohydrodynamics based on staggered-meshed constrained transport with elliptic cleaning," 2021, arXiv:2110.03732 (cit. on pp. 24, 25, 153).

[121] L. Pareschi and G. Russo, "Implicit-explicit runge-kutta schemes and applications to hyperbolic systems with relaxation," Journal of Scientific Computing **25**, 10.1007/BF02728986 (2005) (cit. on pp. 24, 153).

[122] S. R. Lau, G. Lovelace, and H. P. Pfeiffer, "Implicit-explicit (IMEX) evolution of single black holes," Phys. Rev. D **84**, 084023 (2011), arXiv:1105.3922 (cit. on pp. 25, 106, 153).

[123] B. Ripperda et al., "General relativistic resistive magnetohydrodynamics with robust primitive variable recovery for accretion disk simulations," Astrophys. J. Suppl. **244**, 10 (2019), arXiv:1907.07197 (cit. on pp. 25, 153).

[124] W. H. Press et al., *Numerical recipes, The art of scientific computing*, 3rd ed. (Cambridge University Press, Cambridge, England, 2007) (cit. on pp. 25, 79).

[125] J. P. Boyd, *Chebyshev and fourier spectral methods* (Dover Publications Inc, Dover, New York, 2001) (cit. on pp. 27–29).

[126] J. S. Hesthaven and T. Warburton, *Nodal discontinuous Galerkin methods* (Springer, New York, 2008) (cit. on pp. 29–31, 38, 41–43, 46, 48–50, 52, 59, 81, 120).

[127] Y. Saad, *Iterative methods for sparse linear systems*, 2nd ed. (Society for Industrial and Applied Mathematics, 2003) (cit. on pp. 32–34, 80, 81, 93).

[128] T. A. Davis, *Direct methods for sparse linear systems* (Society for Industrial and Applied Mathematics, 2006) (cit. on p. 32).

[129] X. S. Li, "An overview of SuperLU: algorithms, implementation, and user interface," **31**, 302 (2005) (cit. on p. 33).

[130] P. Amestoy et al., "A fully asynchronous multifrontal solver using distributed dynamic scheduling," SIAM Journal on Matrix Analysis and Applications **23**, 15 (2001) (cit. on p. 33).

[131] T. A. Davis, "Algorithm 832: umfpack v4.3—an unsymmetric-pattern multifrontal method," ACM Trans. Math. Softw. **30**, 196 (2004) (cit. on p. 33).

[132] P. Virtanen et al., "SciPy 1.0: fundamental algorithms for scientific computing in python," Nature Methods **17**, 261 (2020) (cit. on p. 33).

[133] P. Grandclément, "KADATH: a spectral solver for theoretical physics," J. Comput. Phys. **229**, 3334 (2010) (cit. on pp. 33, 70, 77, 108).

[134] L. S. Blackford et al., *ScaLAPACK users' guide* (Society for Industrial and Applied Mathematics, Philadelphia, PA, 1997) (cit. on p. 33).

[135] Y. Saad and M. H. Schultz, "GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems," SIAM J. Sci. Stat. Comput. **7**, 856 (1986) (cit. on pp. 33, 55, 80).

[136] S. Balay et al., *PETSc*, `https://www.mcs.anl.gov/petsc`, 2021 (cit. on pp. 34, 79, 144).

[137] Trilinos, *The Trilinos Project*, `https://trilinos.github.io` (visited on 01/05/2022) (cit. on p. 34).

[138] R. D. Falgout, J. E. Jones, and U. M. Yang, "The design and implementation of hypre, a library of parallel high performance preconditioners," in Numerical solution of partial differential equations on parallel computers (2006), pp. 267–294 (cit. on pp. 34, 144).

[139] H. P. Pfeiffer et al., "A multidomain spectral method for solving elliptic equations," Comput. Phys. Commun. **152**, 253 (2003) (cit. on pp. 34, 66, 70, 89, 100, 102, 108, 116).

[140] P. Thoman et al., "A taxonomy of task-based parallel programming technologies for high-performance computing," The Journal of Supercomputing **74**, 1422 (2018) (cit. on p. 35).

[141] L. Kale et al., *The Charm++ parallel programming system*, `https://charm.cs.illinois.edu`, Aug. 2019 (cit. on pp. 35, 71, 78, 106, 137).

[142] L. E. Kidder et al., "SpECTRE: a task-based discontinuous Galerkin code for relativistic astrophysics," J. Comput. Phys. **335**, 84 (2017), arXiv:`1609.00098` (cit. on pp. 35, 38, 41, 71, 78, 79, 137).

[143] W. H. Reed and T. R. Hill, *Triangular mesh methods for the neutron transport equation*, tech. rep. LA-UR-73-479 (Los Alamos Scientific Lab., N. Mex.(USA), 1973) (cit. on p. 38).

[144] B. Cockburn, G. E. Karniadakis, and C.-W. Shu, "The development of discontinuous Galerkin methods," in Discontinuous Galerkin methods (2000), pp. 3–50 (cit. on p. 38).

[145] B. Cockburn, "Devising discontinuous Galerkin methods for non-linear hyperbolic conservation laws," J. Comput. Appl. Math. **128**, 187 (2001) (cit. on p. 38).

[146] M. F. Wheeler, "An elliptic Collocation-Finite element method with interior penalties," SIAM J. Numer. Anal. **15**, 152 (1978) (cit. on p. 38).

[147] D. Arnold et al., "Unified analysis of discontinuous Galerkin methods for elliptic problems," SIAM Journal on Numerical Analysis **39**, 1749 (2002) (cit. on pp. 38, 46, 48, 49).

[148] D. Schötzau et al., "Exponential convergence of hp-DGFEM for elliptic problems in polyhedral domains," in Spectral and high order methods for partial differential equations - ICOSAHOM 2012 (2014), pp. 57–73 (cit. on p. 38).

[149] P. Hansbo and M. G. Larson, "Discontinuous Galerkin methods for incompressible and nearly incompressible elasticity by Nitsche's method," Comput. Methods Appl. Mech. Eng. **191**, 1895 (2002) (cit. on p. 38).

[150] B. Cockburn, D. Schötzau, and J. Wang, "Discontinuous Galerkin methods for incompressible elastic materials," Computer Methods in Applied Mechanics and Engineering **195**, 3184 (2006) (cit. on p. 38).

[151] C. Ortner and E. Süli, "Discontinuous galerkin finite element approximation of nonlinear Second-Order elliptic and hyperbolic systems," SIAM Journal on Numerical Analysis **45**, 1370 (2007) (cit. on p. 38).

[152] S. A. Teukolsky, "Formulation of discontinuous Galerkin methods for relativistic astrophysics," J. Comput. Phys. **312**, 333 (2016), arXiv:1510.01190 (cit. on pp. 38, 40, 41, 52, 53).

[153] F. Fambri et al., "ADER discontinuous Galerkin schemes for general-relativistic ideal magnetohydrodynamics," Mon. Not. Roy. Astron. Soc. **477**, 4543 (2018), arXiv:1801.02839 (cit. on p. 38).

[154] M. Feistauer, F. Roskovec, and A.-M. Sändig, "Discontinuous Galerkin method for an elliptic problem with nonlinear Newton boundary conditions in a polygon," IMA J. Numer. Anal. **39**, 423 (2019) (cit. on pp. 38, 63).

[155] R. Hartmann and P. Houston, "An optimal order interior penalty discontinuous Galerkin discretization of the compressible Navier–Stokes equations," J. Comput. Phys. **227**, 9670 (2008) (cit. on pp. 38, 50).

[156] Y. Epshteyn and B. Rivière, "Estimation of penalty parameters for symmetric interior penalty Galerkin methods," J. Comput. Appl. Math. **206**, 843 (2007) (cit. on pp. 38, 50).

[157] J. D. De Basabe, M. K. Sen, and M. F. Wheeler, "The interior penalty discontinuous Galerkin method for elastic wave propagation: grid dispersion," Geophys. J. Int. **175**, 83 (2008) (cit. on pp. 38, 50).

[158] D. Fortunato, C. H. Rycroft, and R. Saye, "Efficient operator-coarsening multigrid schemes for local discontinuous Galerkin methods," SIAM J. Sci. Comput. **41**, A3913 (2019) (cit. on pp. 41, 46, 55, 57, 82).

[159] D. A. Kopriva, *Implementing spectral methods for partial differential equations* (Springer, 2009) (cit. on pp. 42, 52).

[160] J. Peraire and P.-O. Persson, "The compact discontinuous galerkin (cdg) method for elliptic problems," SIAM J. Sci. Comput. **30**, 1806 (2008), arXiv:math/0702353 (cit. on pp. 48, 49).

[161] J. Douglas and T. Dupont, "Interior penalty procedures for elliptic and parabolic Galerkin methods," in Computing methods in applied sciences (1976), pp. 207–216 (cit. on p. 48).

[162] K. Shahbazi, "An explicit expression for the penalty parameter of the interior penalty method," J. Comput. Phys. **205**, 401 (2005) (cit. on pp. 49, 81).

[163] K. Hillewaert, *Development of the discontinuous Galerkin method for high-resolution, large scale CFD and acoustics in industrial geometries* (Presses univ. de Louvain, Feb. 2013) (cit. on pp. 49, 50).

[164] S. A. Teukolsky, "Short note on the mass matrix for Gauss-Lobatto grid points," J. Comput. Phys. **283**, 408 (2015), arXiv:1412.2276 (cit. on p. 52).

[165] G. Mengaldo et al., "Dealiasing techniques for high-order spectral element methods on regular and irregular grids," J. Comput. Phys. **299**, 56 (2015) (cit. on p. 53).

[166] Y. Levin, "Internal thermal noise in the LIGO test masses: a direct approach," Phys. Rev. D **57**, 659 (1998), arXiv:gr-qc/9707013 (cit. on pp. 59, 136, 137).

[167] G. Lovelace, N. Demos, and H. Khan, "Numerically modeling Brownian thermal noise in amorphous and crystalline thin coatings," Class. Quantum Gravity **35**, 025017 (2018), arXiv:1707.07774 (cit. on pp. 59, 60, 104, 136, 137, 144, 145, 147, 148).

[168] Y. T. Liu and K. S. Thorne, "Thermoelastic noise and homogeneous thermal noise in finite sized gravitational wave test masses," Phys. Rev. D **62**, 122002 (2000), arXiv:gr-qc/0002055 (cit. on p. 59).

[169] G. Lovelace, "The dependence of test-mass coating and substrate thermal noise on beam shape in the advanced laser interferometer gravitational-wave observatory (advanced LIGO)," Class. Quantum Gravity **24**, 4491 (2007), arXiv:gr-qc/0610041 (cit. on p. 59).

[170] H. P. Pfeiffer, "Initial data for black hole evolutions," PhD thesis (Cornell University, 2003), arXiv:gr-qc/0510016 (cit. on pp. 64, 113, 116–118, 124).

[171] *Supplemental material: input-file configurations to reproduce the results presented in this article with the spectre code*, arXiv:2108.05826/anc (cit. on p. 66).

[172] J. D. Hunter, "Matplotlib: a 2d graphics environment," Comput. Sci. Eng, **9**, 90 (2007) (cit. on pp. 67, 106, 149).

[173] T. A. Caswell et al., *matplotlib*, 10.5281/zenodo.4268928, version v3.3.3, Nov. 2020 (cit. on pp. 67, 106, 149).

[174] T. Tantau, *pgf - a portable graphic format for TeX*, github:pgf-tikz/pgf, 2021 (cit. on pp. 67, 106, 149).

[175] J. Ahrens, B. Geveci, and C. Law, "ParaView: an end-user tool for large-data visualization," in Visualization handbook (2005) (cit. on pp. 67, 106, 149, 157).

[176] G. B. Cook, "Initial data for numerical relativity," Living Rev. Relativity **3**, 10.12942/lrr-2000-5 (2000), arXiv:gr-qc/0007085 (cit. on p. 70).

[177] R. Abbott et al. (LIGO, Virgo, KAGRA), "Observation of gravitational waves from two neutron star–black hole coalescences," Astrophys. J. Lett. **915**, L5 (2021), arXiv:2106.15163 (cit. on p. 70).

[178] E. Gourgoulhon et al., *LORENE*, http://www.lorene.obspm.fr (cit. on p. 70).

[179] P. Grandclement, "Accurate and realistic initial data for black hole-neutron star binaries," Phys. Rev. D **74**, 124002 (2006), arXiv:gr-qc/0609044 (cit. on p. 70).

[180] M. Ansorg, "A double-domain spectral method for black hole excision data," Phys. Rev. D **72**, 024018 (2005), arXiv:gr-qc/0505059 (cit. on p. 70).

[181] S. Ossokine et al., "Improvements to the construction of binary black hole initial data," Class. Quantum Gravity **32**, 245010 (2015), arXiv:1506.01689 (cit. on pp. 70, 116, 122).

[182] F. Foucart et al., "Initial data for black hole-neutron star binaries: a flexible, high-accuracy spectral method," Phys. Rev. D **77**, 124051 (2008), arXiv:0804.3787 (cit. on pp. 70, 131).

[183] W. Tichy et al., "Constructing binary neutron star initial data with high spins, high compactnesses, and high mass ratios," Phys. Rev. D **100**, 124046 (2019), arXiv:1910.09690 (cit. on pp. 70, 134).

[184] K. Uryū and A. Tsokaros, "New code for equilibriums and quasiequilibrium initial data of compact objects," Phys. Rev. D **85**, 064014 (2012), arXiv:1108.3065 (cit. on p. 70).

[185] T. Assumpcao, L. R. Werneck, T. P. Jacques, et al., "NRPyElliptic: a fast hyperbolic relaxation elliptic solver for numerical relativity, I: conformally flat, binary puncture initial data," 2021, arXiv:2111.02424 (cit. on p. 70).

[186] H. R. Rüter et al., "Hyperbolic relaxation method for elliptic equations," Phys. Rev. D **98**, 10.1103/PhysRevD.98.084044 (2018), arXiv:1708.07358 (cit. on p. 70).

[187] E. Schnetter, *CarpetX*, 10.5281/zenodo.6131528, 2022 (cit. on p. 71).

[188] W. Zhang et al., "AMReX: a framework for block-structured adaptive mesh refinement," J. Open Source Softw. **4**, 1370 (2019) (cit. on p. 71).

[189] M. Fernando et al., "Massively parallel simulations of binary black hole intermediate-mass-ratio inspirals," SIAM J. Sci. Comput. **41**, C97 (2019), arXiv:1807.06128 (cit. on p. 71).

[190] W. Tichy, A. Adhikari, and L. Ji, "Numerical relativity with the new Nmesh code," Bull. Am. Phys. Soc. **65** (2020) (cit. on p. 71).

[191] M. Bugner et al., "Solving 3D relativistic hydrodynamical problems with weighted essentially nonoscillatory discontinuous Galerkin methods," Phys. Rev. D **94**, 084004 (2016), arXiv:1508.07147 (cit. on p. 71).

[192] B. Daszuta et al., "GRAthena++: puncture evolutions on vertex-centered oct-tree adaptive mesh refinement," Astrophys. J. Supp. **257**, 25 (2021), arXiv:2101.08289 (cit. on p. 71).

[193] A. Reinarz et al., "ExaHyPE: an engine for parallel dynamically adaptive simulations of wave problems," Comput. Phys. Commun. **254**, 107251 (2020) (cit. on p. 71).

[194] H. Sagan, *Space-filling curves* (Springer, New York, NY, 1994) (cit. on p. 77).

[195] R. Borrell et al., "Parallel mesh partitioning based on space filling curves," Comput. Fluids **173**, 264 (2018) (cit. on p. 78).

[196] J. E. Dennis Jr. and R. B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations* (SIAM, Philadelphia, 1996) (cit. on p. 79).

[197] P. R. Brune et al., "Composing scalable nonlinear algebraic solvers," SIAM Rev. **57**, 535 (2015), arXiv:1607.04254 (cit. on p. 79).

[198] W. L. Briggs, V. E. Henson, and S. F. McCormick, *A multigrid tutorial*, 2nd ed. (SIAM, Philadelphia, 2000) (cit. on pp. 81, 105).

[199] J. W. Lottes and P. F. Fischer, "Hybrid Multigrid/Schwarz algorithms for the spectral element method," J. Sci. Comput. **24**, 45 (2005) (cit. on pp. 84, 91).

[200] J. Stiller, "Robust multigrid for high-order discontinuous Galerkin methods: a fast Poisson solver suitable for high-aspect ratio Cartesian grids," J. Comput. Phys. **327**, 10.1016/j.jcp.2016.09.041 (2016), arXiv:1603.02524 (cit. on pp. 84–87, 89, 91).

[201] A. AlOnazi, G. S. Markomanolis, and D. Keyes, "Asynchronous task-based parallelization of algebraic multigrid," in Proceedings of the platform for advanced scientific computing conference, PASC '17 (2017) (cit. on pp. 84, 99, 105).

[202] K. Kang, "Scalable implementation of the parallel multigrid method on massively parallel computers," Comput. Math. Appl. **70**, 2701 (2015) (cit. on p. 84).

[203] G. Guennebaud, B. Jacob, et al., *Eigen*, http://eigen.tuxfamily.org, version 3.4, 2021 (cit. on p. 90).

[204] Y. Saad, "ILUT: a dual threshold incomplete LU factorization," Numer. Linear Algebra Appl. **1**, 387 (1994) (cit. on p. 91).

[205] *Supplemental material: input-file configurations to reproduce the results presented in this article with the spectre code*, arXiv:2111.06767/anc (cit. on p. 104).

[206] B. Cockburn, J. Gopalakrishnan, and R. D. Lazarov, "Unified hybridization of discontinuous Galerkin, mixed, and continuous Galerkin methods for second order elliptic problems," SIAM J. Numer. Anal. **47**, 1319 (2009) (cit. on p. 104).

[207] M. Giacomini, R. Sevilla, and A. Huerta, "HDGlab: an open-source implementation of the hybridisable discontinuous Galerkin method in MATLAB," Arch. Comput. Methods Eng. **28**, 1941 (2021) (cit. on p. 104).

[208] S. Muralikrishnan, T. Bui-Thanh, and J. N. Shadid, "A multilevel approach for trace system in HDG discretizations," J. Comput. Phys. **407**, 109240 (2020) (cit. on p. 104).

[209] L. T. Buchman et al., "Simulations of unequal-mass black hole binaries with spectral methods," Phys. Rev. D **86**, 084033 (2012) (cit. on p. 104).

[210] S. Markidis, "The old and the new: can physics-informed deep-learning replace traditional linear solvers?" Front. big data **4**, 10.3389/fdata.2021.669097 (2021), arXiv:2103.09655 (cit. on p. 105).

[211] V. Guidetti et al., "dNNsolve: an efficient NN-based PDE solver," 2021, arXiv:2103.08662 (cit. on p. 105).

[212] H. Deng, L. Mayer, and H. Latter, "Global simulations of self-gravitating magnetized protoplanetary disks," Astrophys. J. **891**, 154 (2020) (cit. on pp. 106, 153).

[213] P. F. Hopkins, "A new class of accurate, mesh-free hydrodynamic simulation methods," Mon. Not. Roy. Astron. Soc. **450**, 53 (2015), arXiv:1409.7395 (cit. on pp. 106, 153).

[214] G. L. Bryan et al. (Enzo), "Enzo: an adaptive mesh refinement code for astrophysics," Astrophys. J., Suppl. Ser. **211**, 19 (2014) (cit. on pp. 106, 153).

[215] M. Visser, "The Kerr spacetime: a brief introduction," in Kerr Fest: black holes in astrophysics, general relativity and quantum gravity (June 2007), arXiv:0706.0622 (cit. on pp. 110, 112).

[216] T. W. Baumgarte et al., "Implementing an apparent horizon finder in three-dimensions," Phys. Rev. D **54**, 4849 (1996), arXiv:gr-qc/9606010 (cit. on p. 115).

[217] R. Owen, "The final remnant of binary black hole mergers: multipolar analysis," Phys. Rev. D **80**, 084012 (2009), arXiv:0907.0280 (cit. on p. 115).

[218] R. Owen et al., "Black hole spin axis in numerical relativity," Phys. Rev. D **99**, 084031 (2019), arXiv:1708.07325 (cit. on p. 115).

[219] H. P. Pfeiffer et al., "Reducing orbital eccentricity in binary black hole simulations," Class. Quant. Grav. **24**, edited by M. Campanelli and L. Rezzolla, S59 (2007), arXiv:gr-qc/0702106 (cit. on p. 116).

[220] A. Buonanno et al., "Reducing orbital eccentricity of precessing black-hole binaries," Phys. Rev. D **83**, 104034 (2011), arXiv:1012.1549 (cit. on p. 116).

[221] A. H. Mroue and H. P. Pfeiffer, "Precessing binary black holes simulations: quasicircular initial data," (2012), arXiv:1210.2958 (cit. on p. 116).

[222] G. B. Cook and M. A. Scheel, "Well-behaved harmonic time slices of a charged, rotating, boosted black hole," Phys. Rev. D **56**, 4775 (1997) (cit. on p. 117).

[223] **N. L. Vu** et al., *Binary black hole initial data example in the spectre documentation*, https://spectre-code.org/example_bbh_id.html (visited on 03/05/2022) (cit. on p. 117).

[224] B. Szilágyi, "Key elements of robustness in binary black hole evolutions using spectral methods," Int. J. Mod. Phys. D **23**, 1430014 (2014), arXiv:1405.3693 (cit. on p. 120).

[225] L. Lindblom, "Phase transitions and the mass-radius curves of relativistic stars," Physical Review D **58**, 10.1103/PhysRevD.58.024008 (1998) (cit. on pp. 126–128).

[226] T. W. Baumgarte, *Oppenheimer-Volkov solution in isotropic coordinates*, (2009) https://einsteintoolkit.org/thornguide/EinsteinInitialData/TOVSolver/documentation.html (cit. on pp. 128, 129).

[227] C. W. Misner, K. S. Thorne, and J. A. Wheeler, *Gravitation* (W. Freeman, 1973) (cit. on p. 129).

[228] T. W. Baumgarte, N. Ó Murchadha, and H. P. Pfeiffer, "The Einstein constraints: uniqueness and non-uniqueness in the conformal thin sandwich approach," Phys. Rev. **D75**, 044009 (2007), arXiv:gr-qc/0610120 (cit. on pp. 130, 132).

[229] R. H. Price, C. Markakis, and J. L. Friedman, "Iteration stability for simple Newtonian stellar systems," J. Math. Phys. **50**, 073505 (2009), arXiv:0903.3074 (cit. on p. 134).

[230] X. Huang et al., "Quasiequilibrium models for triaxially deformed rotating compact stars," Phys. Rev. D **78**, 124023 (2008) (cit. on p. 134).

[231] K. Henriksson et al., "Initial data for high-compactness black hole–neutron star binaries," Class. Quant. Grav. **33**, 105009 (2016), arXiv:1409.7159 (cit. on p. 134).

[232] G. D. Cole et al., "Tenfold reduction of Brownian noise in high-reflectivity optical coatings," Nat. Photonics **7**, 644 (2013) (cit. on p. 136).

[233] H. B. Callen and T. A. Welton, "Irreversibility and generalized noise," Physical Review **83**, 34 (1951) (cit. on p. 136).

[234] W. Bernard and H. B. Callen, "Irreversible thermodynamics of nonlinear processes and noise in driven systems," Rev. Mod. Phys. **31**, 1017 (1959) (cit. on p. 136).

[235] R. Kubo, "The fluctuation-dissipation theorem," Reports on progress in physics **29**, 255 (1966) (cit. on p. 136).

[236] G. M. Harry et al., "Thermal noise in interferometric gravitational wave detectors due to dielectric optical coatings," Classical and Quantum Gravity **19**, 897 (2002) (cit. on p. 139).

[237] T. Hong et al., "Brownian thermal noise in multilayer coated mirrors," Phys. Rev. D **87**, 082001 (2013) (cit. on p. 139).

[238] D. Arndt et al., "The `deal.II` library," Journal of Numerical Mathematics (2021) (cit. on pp. 144, 149).

[239] D. Arndt et al., "The deal.II finite element library: design, features, and insights," Computers & Mathematics with Applications **81**, 407 (2021), arXiv:`1910.13247` (cit. on pp. 144, 149).

[240] G. Venugopalan, K. Arai, and R. X. Adhikari, *Global optimization of multilayer dielectric coatings for precision measurements*, 2021, arXiv:`2110.13437` (cit. on pp. 149, 152).

[241] A. Riols and H. Latter, "Gravitoturbulent dynamos in astrophysical discs," Mon. Not. Roy. Astron. Soc. **482**, 3989 (2019), arXiv:`1810.06589` (cit. on p. 153).

[242] **N. L. Fischer** et al., *Visualization of the GW190412 event*, version v5, Apr. 2020 (cit. on p. 157).

[243] **N. L. Fischer** et al., *Visualization of the GW190814 event*, version v3, June 2020 (cit. on pp. 157, 158).

[244] **N. L. Fischer**, H. P. Pfeiffer, and A. Buonanno, *Visualization of the GW190521 event*, version v1, Sept. 2020 (cit. on p. 157).

[245] T. Dietrich et al., *Simulation of an NSBH coalescence consistent with GW200115*, `https://www.ligo.caltech.edu/WA/video/ligo20210629v4` (visited on 05/26/2022) (cit. on p. 157).

[246] M. Boyle, *Python package* `spherical`, 10.5281/zenodo.4045222, 2022 (cit. on p. 158).

[247] A. Bohn et al., "What does a binary black hole merger look like?" Class. Quant. Grav. **32**, 065002 (2015), arXiv:`1410.7775` (cit. on p. 159).

Acknowledgements

First and foremost, I would like to thank Harald Pfeiffer for advising me during these past four years, and for teaching me the dark art that is composing preconditioners. His ability to glance at a plot and tell you what's wrong in your code has both scared and helped me on multiple occasions. I appreciate the freedom he gave me to pursue my strengths, while being always available to offer practical advice and guidance. I pledge to always make convergence plots. I also thank Alessandra Buonanno and Masaru Shibata for their advice as my second advisor and mentor during this time.

I am grateful to the ACR family at AEI Potsdam for countless controversial discussions over coffee, and for humiliating me in chess. In particular, I thank my fellow Italian and non-Italian Ph.D. students for making the time at the AEI so absolutely enjoyable: Andrea Antonelli, Riccardo Barbieri, Ollie Burke, Roberto Cotesta, Mohammed Khalil, Lorenzo Pompili, Luca Prudenzi, Stefano Savastano, Noah Sennett, Lorenzo Speri, and also Niko Wittek.

I also extend my gratitude to my second academic family, the SXS collaboration. It is a great pleasure to work with a crowd of people who share my ambition to write decent scientific software. I look forward to working with everyone more directly than across the Atlantic when I join Caltech in the Fall.

Thank you to Prof. Helvi Witek and Prof. Bernd Brügmann for agreeing to referee my thesis.

Finally, I want to thank my family and friends. I am especially grateful for the newest member of my family: my son Charlie, born only three weeks before writing these words. He hasn't really made it any easier to complete this thesis, but meeting him has been the greatest joy of my life. I also thank my wife Trang for accompanying me for all this time, and in particular on our next adventure to California.

Figure C.1: Charlie Vu, born May 3, 2022.