Implementation and Technical Developments of ExaHyPE2 for Astrophysics Applications

HAN ZHANG

ZHANG, HAN (2023) Implementation and Technical Developments of ExaHyPE2 for Astrophysics Applications. Doctoral thesis, Durham University.

Implementation and Technical Developments of ExaHyPE2 for Astrophysics Applications

Han Zhang

A Thesis presented for the degree of
Doctor of Philosophy

Institute for Computational Cosmology
Department of Physics
Durham University
United Kingdom
July 2023

Abstract

This thesis presents the two astrophysics applications we develop on the code platform ExaHyPE2, along with the technical advancement we achieved during their implementation.

Our first application focuses on simulating the spherical accretion of collisional gas, considering both the standard and a specific extension of Dvali-Gabadadze-Porrati (DGP) gravity. The spherical accretion scenario has significant importance in cosmology since it captures several crucial aspects of structure formation. While it has been widely studied in standard gravity, we discover a self-similar solution under modified gravity for the first time in this scenario. The application successfully reproduces the theoretical prediction and the self-similarity is observed under both standard and modified gravity in our simulations. This good agreement confirms the reliability of our application in modelling astrophysical processes in the spherical collapse scenario.

Our second application is a complete numerical relativity code designed for astrophysics scenarios in black hole spacetimes. Numerical relativity is a widely utilized approach in simulating astrophysics systems in the strong field regions of gravity and is also used in extracting gravitational wave signals from them. In this thesis, we report the theoretical background, detailed code implementations and simulation tests of our code. The results demonstrate that our code can effectively perform simulations in the black hole spacetime, and exhibit stable evolutions in the dynamics systems, including the rotating binary black hole mergers. Furthermore, we can properly extract the gravitational wave signal from the domain in our tests. However, we have also observed some instability affecting the long-term evolution in our simulations, which has become the focus of our ongoing investigation.

In addition to reporting our astrophysics applications, we introduce the powerful partial derivative equations (PDEs) solving engine ExaHyPE2 in this thesis. This engine offers a relatively flexible code structure, allowing users to implement simulations according to their specific needs. In this regard, this thesis describes the kernel computational solvers, the refinement transition strategy, the radiative boundary condition and the particle module, which are all new features developed during the implementation of our astrophysics code. These advancements enhance the overall capability of ExaHyPE2 to address various problems across different branches of science and engineering in the future.

Supervisors:

Prof. Baojiu Li (The Institute for Computational Cosmology, Durham University)
Prof. Tobias Weinzierl (Department of Computer Science, Durham University)

Acknowledgements

I would like to first express my sincere gratitude to my supervisors, Baojiu and Tobias. Both of them are esteemed experts and researchers in their fields. Developing new simulation codes is quite challenging work, and without their profound knowledge, insightful feedback, and practical support during the research, this project would not have become a reality. I am truly grateful for the privilege of working with them throughout the entire PhD study.

My thanks also go to the collaborators during the code development: Cristian, Anne, Francesco and Dominic, as well as other researchers from Tobias' team. Their constructive discussions and suggestions help us to delve deeper into the code engine and application, and understand more in technical details of implementing the numerical relativity system on ExaHyPE2.

I would like to thank Katy and her postdoc Lorenzo, who are from the research group of GRchombo. The shared insights, explanations and sample data from their side are invaluable in developing our newborn code.

My thanks are extended to Alastair and the entire technical support team of COSMA. Nearly all the simulations during this research project were conducted on the COSMA cluster. Their hard work ensured that our developing code has a solid

and stable platform for tests and production.

I am very grateful for the financial support provided by the joint studentship of the China Scholarship Council and Durham University. Their general funding allows me to focus on my research and pursue my academic career without financial concerns.

I would like to express my gratitude to my family, for their love, encouragement and belief in me during my PhD study. To my friends, thank you for being there on these challenging days and you make the journey more enjoyable. A special thank you goes to my girlfriend, Liang Yang, for her constant and gentle accompany over the past two years. Without them, I would not have had enough energy and determination to overcome the difficulties and complete my PhD.

The work conducted in this thesis used the DiRAC@Durham facility managed by the Institute for Computational Cosmology on behalf of the STFC DiRAC HPC Facility (`www.dirac.ac.uk`). The equipment was funded by BEIS via STFC capital grants ST/K00042X/1, ST/P002293/1, ST/R002371/1 and ST/S002502/1, Durham University and STFC operation grant ST/R000832/1. DiRAC is part of the UK National e-Infrastructure. ExaHyPE2 is currently maintained and extended as part of the embedded CSE programme of the ARCHER2 UK National Supercomputing Service (`http://www.archer2.ac.uk`) under grant ARCHER2-eCSE04-2 and Durham's oneAPI Academic Centre of Excellence made by Intel.

Declaration

The work in this thesis is based on research carried out at the Institute for Computational Cosmology in Department of Physics, Durham University, United Kingdom. No part of this thesis has been submitted elsewhere for any other degree or qualification and it is all my own work unless referenced to the contrary in the text.

A part of the content in this thesis has appeared in the following papers:

Han Zhang et al. "Spherical accretion of collisional gas in modified gravity I: self-similar solutions and a new cosmological hydrodynamical code". In: *Monthly Notices of the Royal Astronomical Society* 515.2 (2022), pp. 2464–2482.

Li, B., Schulz, H., Weinzierl, T., **Zhang, H.** (2022). "Dynamic Task Fusion for a Block-Structured Finite Volume Solver over a Dynamically Adaptive Mesh with Local Time Stepping". In: Varbanescu, AL., Bhatele, A., Luszczek, P., Marc, B. (eds) *High Performance Computing. ISC High Performance 2022. Lecture Notes in Computer Science*, vol 13289. Springer, Cham.

Contents

List of Figures

CHAPTER 1

Introduction

1.1 Numerical Simulation in Astrophysics

Over the past three decades, the field of astrophysics has witnessed significant advancements in research based on numerical simulations. With the aid of increasingly powerful computer clusters, scientists can now model and explore a wide range of astrophysics phenomena in a computation domain, from the dynamics of black holes to the formation of galaxies and stars and the evolution of the universe itself. Those phenomena are quite complex and often coupled with extreme conditions, such as high temperature, large velocity and strong gravity, making them challenging to study solely through direct observation or theoretical analysis. Fortunately, numerical simulation now offers a third approach to investigating those phenomena, giving valuable insights into the underlying physics.

Numerical simulation gains its importance for bridging the gap between observations and theories in astrophysics. On the one hand, theorists can implement different theoretical models and hypotheses in simulations, especially for those astrophysics phenomena that have limited observational data or occur infrequently. By comparing the theoretical predictions, the simulation results and observation data, theorists can examine their models directly and gain insights into their behaviours. On the other hand, observers can utilize simulation results to estimate the physical quantities associated with observed data, which sometimes are difficult or even impossible to obtain through analytical calculations alone. Moreover, when new theories do not produce directly observable predictions through analytical calculations, predictions by numerical simulations can also guide future observations and experiments.

There is a vast body of literature work that contributes to the development of the field of numerical simulations in astrophysics. Due to the very broad scope of this field Bodenheimer et al., 2006, e.g., it is challenging to review them in this thesis. Therefore, here we will focus on the two representative scenarios in astrophysics that we develop applications for: the spherical accretion of gas, and the evolution of black hole systems. The former is a useful tool to understand how gas evolves in a spherical system that mimics typical galaxy-forming regions and hence can provide useful insight into the physics of galaxy formation. The latter is indispensable in the study of black holes and the gravitational waves emitted by them.

There have been a significant number of open-source numerical simulation codes for various astrophysics phenomena in the community which have yielded important results in the studies of those scenarios. However, there is still an ongoing need for the development of new codes that offer higher efficiency and accuracy for more complicated astrophysics processes. By incorporating state-of-the-art algorithms and improved parallelization techniques, new codes can offer the possibility to perform simulations on more realistic scenarios and address previously inaccessible regimes. In line with this goal, this thesis reports the brand-new code applications we developed on the new platform ExaHyPE2, for the two specific astrophysics scenarios mentioned above, they are described in the next two subsections in detail.

1.1.1 Spherical Accretion in Modified Gravity

Spherical collapse (accretion) is a widely studied phenomenon in cosmology. It describes the evolution of a spherically symmetric overdense region (see Figure 1.1): how it decouples from the Hubble flow, turns around, and finally collapses into a

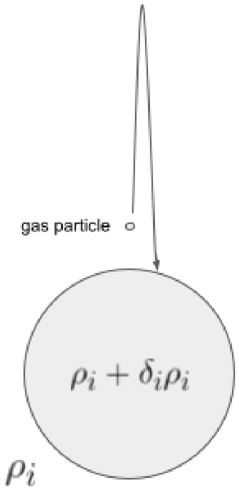

Figure 1.1: The illustration of the spherical collapse. An overdense region with a density of $\rho_i + \delta_i \rho_i$ is present in the uniform background with the density of ρ_i. The gas particles, which follow the expanding Hubble flow initially, will decelerate, turns around and finally collapses back to the overdense region due the gravity.

singularity or some virialised matter distribution. Despite its simplicity, this scenario is of great importance, as it can describe several crucial aspects of structure formation of different matter components in our Universe (e.g., collisionless dark matter and collisional baryonic gas), thus providing valuable insights into the real and more complicated cosmological process of galaxy formation. Some cosmological hydrodynamical simulation codes also adopt this scenario as a test of their reliability and accuracy(e.g., RAMSES, Teyssier, 2002), because in certain cases analytical solutions can be found, as I will review later on.

The study of spherical collapse has a long history, with some of the early works including Gunn and Gott (1972), Fillmore and Goldreich (1984), Ryden and Gunn (1987), and Subramanian, Cen, and Ostriker (2000). Among them, Bertschinger, 1985 revealed an elegant self-similarity in the solution for a matter-dominated, Einstein-de Sitter (EdS), universe, for both collisionless and collisional matter. Using the turnaround radius, $r_{\mathrm{ta}}(t)$, in the EdS model, the various quantities in the system of evolution equations can be rescaled, such that all the dependencies on the spherical radius r and time t are reduced into the dependence on a single variable $\lambda \equiv r/r_{\mathrm{ta}}(t)$. This gives a unique set of solutions of physical quantities, expressed in terms of λ, which can be used to obtain the status of the evolution at arbitrary (r, t). Spherical collapse is one of the few scenarios where a detailed semi-analytical solution is known in cosmology.

In the past decades, a lot of effort has been made to incorporate more physical processes into the spherical collapse model. Based on the original radial collapse of matter, there are studies that look into the effects of angular momentum (Ryden, 1988; Sikivie, Tkachev, and Wang, 1997; Le Delliou and Henriksen, 2003), dynamical friction (Antonuccio-Delogu and Colafrancesco, 1994; Popolo, 2009) and shears (Del Popolo, Pace, and Lima, 2013; Pace, Batista, and Del Popolo, 2014). On the thermodynamics side, several research also studies the cooling and heating process during the collapse (Abadi, Bower, and Navarro, 2000; Uchida and Yoshida, 2004; McCarthy et al., 2007) for a more realistic thermal history of the gas.

Nowadays, studies of cosmological structure formation have entered a highly advanced stage, with more complicated physical processes added into increasingly sophisticated hydrodynamical simulations (e,g, Schaye et al., 2015; McCarthy et al., 2017; Springel et al., 2018), which can realistically reproduce the observed properties of galaxies clusters; see, e.g., Borgani and Kravtsov (2011), for a review. Comparatively, therefore, the role of spherical collapse as a stand-alone simulation experiment has declined. However, this scenario can still serve as a useful benchmark test to assess the accuracy and reliability of new simulation codes. This partially explains why the original spherical collapse model with self-similarity still attracts attention (e.g., Halle, Colombi, and Peirani, 2019; Alard, 2020). It is also worth noting that the self-similarity can still hold under some other circumstances, if the physics added (e.g., the cooling function) follows certain assumptions (Sikivie, Tkachev, and Wang, 1997; Uchida and Yoshida, 2004).

The application we report in this thesis concerns spherical collapse in modified gravity (MG) models, which are an alternative solution to avoid several problems of the current concordance ΛCDM cosmological model. The ΛCDM model suggests that the majority of energy density in the Universe contributed by cold dark matter (CDM), a species of non-baryonic and non-relativistic particles, and dark energy, an energy component with exotic properties (e.g., negative pressure), in the form of a positive cosmological constant Λ (Amendola and Tsujikawa, 2010), which is needed to explain the accelerated Hubble expansion. However, the hypothetical Λ suffers from long-standing theoretical problems (Weinberg, 1989). Modified gravity models have therefore been proposed as an attempt to overcome those issues by extending the standard General Relativity (GR) rather than assuming extra unknown matter or energy components (e.g., Sotiriou and Faraoni, 2010; Linder, 2010). In recent years, there has been growing interest in MG models, because constraining them using various astrophysical and cosmological observations offers a powerful way to test our theory of gravity.

In this context, there are already various studies which look into the spherical collapse scenarios in different modified gravity models (e.g., Martino, Stabenau, and Sheth, 2009; Schmidt, Hu, and Lima, 2010; Li and Efstathiou, 2012; Lombriser, Koyama, and Li, 2014; Barreira et al., 2014; Lopes et al., 2018; Contigiani, Vardanyan, and Silvestri, 2019). However, these studies generally focus on models that no longer uphold the property of self-similarity. This is not surprising, because even within GR there are strict conditions which must be satisfied to have self-similar solutions. For example, the EdS model loses its self-similarity property once a cosmological constant is added.

In our application, we investigate the spherical collapse scenario for collisional gas in both the Einstein-de Sitter universe and a slightly modified version of the Dvali-Gabadadze-Porrati (DGP, Dvali, Gabadadze, and Porrati, 2000) braneworld model. The latter is a class of MG models that has attracted much attention in the last two decades, featuring an enhanced strength of the total gravitational force and the Vainshtein screening mechanism (Vainshtein, 1972), which suppresses deviations from GR near massive objects to give the model a chance of passing the stringent Solar System and lab constraints. Despite its complexity, we find that the self-similarity property can still be achieved in this model under certain conditions that are not unnatural. Our self-similar solutions in this model will provide insights into how these mechanisms of modified gravity may affect structure formation in similar, but more realistic, models where self-similarity no longer happens.

1.1.2 "The Old Holy Grail": Binary Black Holes Merger in Numerical Relativity

The second astrophysics application we implement in ExaHyPE2 is a numerical relativity code designed for simulations in black hole spacetimes. It is also specifically geared towards studying the process of the binary black hole merger and its gravitational wave signal.

Numerical relativity is essential in simulating black hole mergers, as the considered objects are located in the strong-field regime of gravity, where approximate methods such as the Newtonian method are hard to work and analytical solutions can not be found. Those facts make numerical simulation the only viable approach to studying this scenario. By recasting the Einstein field equation properly, numerical relativity solves the physics system by treating it as a Cauchy initial value problem, which one can solve via standard numerical integral schemes. Currently, there are two leading approaches to cast the Einstein equations, resulting in two different formulations of the evolving system.

The first evolving system is the generalised-harmonic formalism (Lindblom et al., 2006; Szilágyi et al., 2007), which adopts a full four-dimensional form of the field equations with a harmonic coordinate. The principle parts of the systems in this formulation are wave equations with very clear mathematical properties, and the damping terms can be used to control the constraint violations (Alic et al., 2012). However, the generalised-harmonic formalism can not deal with singularities. Therefore excision must be implemented to remove them when using this formalism in black hole simulations (Boyle et al., 2007). The other main formalism is based on the 3+1 foliation of spacetime, which splits time and space explicitly and rewrites the field equations accordingly. The first attempt of this approach was the Arnowitt-Deser-Misner (ADM) formulation developed in the 1960s (Arnowitt, Deser, and Misner, 2008). However, it was shown to be only weakly hyperbolic with the usual gauge choices and therefore is not stable in numerical simulations. At the end of the last century, a new, strong-hyperbolic, formulation derived from the ADM equations was published, which is known as BSSNOK, named after the six researchers who contributed to its proposal (Baumgarte and Shapiro, 1998a; Shibata and Nakamura, 1995a; Nakamura, Oohara, and Kojima, 1987). This is a breakthrough that led to the ability to have long-term stable evolutions of black hole systems. One of the advantages of the 3+1 foliation approach is that it can adapt puncture methods (Brandt and Brügmann, 1997) to deal with singularities without excision, which we will describe in detail in Section 4.1.2. It makes the implementation of these numerical simulation codes much easier when considering spacetimes with singularities.

Besides the BSSNOK, several other strongly-hyperbolic formulations have been proposed in the last twenty years, including the constrained formulation of Bonazzola et al., 2004, and the so-called Z4 family. The original formulation of Z4 was proposed by Bona et al., 2003a, which was further developed into formulations of CCZ4 (Alic et al., 2012) and Z4c (Hilditch et al., 2013). One of the most significant differences of the Z4 formulation from BSSNOK, which is also its advantage is that it introduces an auxiliary four-dimensional vector Z to propagate the constraint violations off the simulation domain, thus further enhancing the stability. Damping terms are also introduced in the Z4 formulation, inspired by the generalised harmonic formulation.

Studies of binary black hole mergers in numerical relativity have a long history and the first investigation with numerical consideration can be traced back to the 1980s (Smarr, 1977). Since then, many researchers devoted years of work to this ”holy grail” problem, aiming to simulate this scenario on computers. However, it was not until the beginning of this century did accurate and long-term simulation of inspiraling binary black holes became possible (Pretorius, 2005, with an illustration of its result shown in figure 1.2). This significant advancement in the field was the result of several breakthroughs during that period, both in techniques and in the understanding of its physics. On the one hand, better formulations of the evolving system in numerical relativity were developed, as we mentioned above, enhancing the accuracy and stability. Strategies on singularities avoidance also got improved both in the black hole excision (Yo, Baumgarte, and Shapiro, 2002; Alcubierre et al., 2005) and puncture approach (Brügmann et al., 2008). On the other hand, more advanced numerical algorithms and better parallelization techniques allow sci-

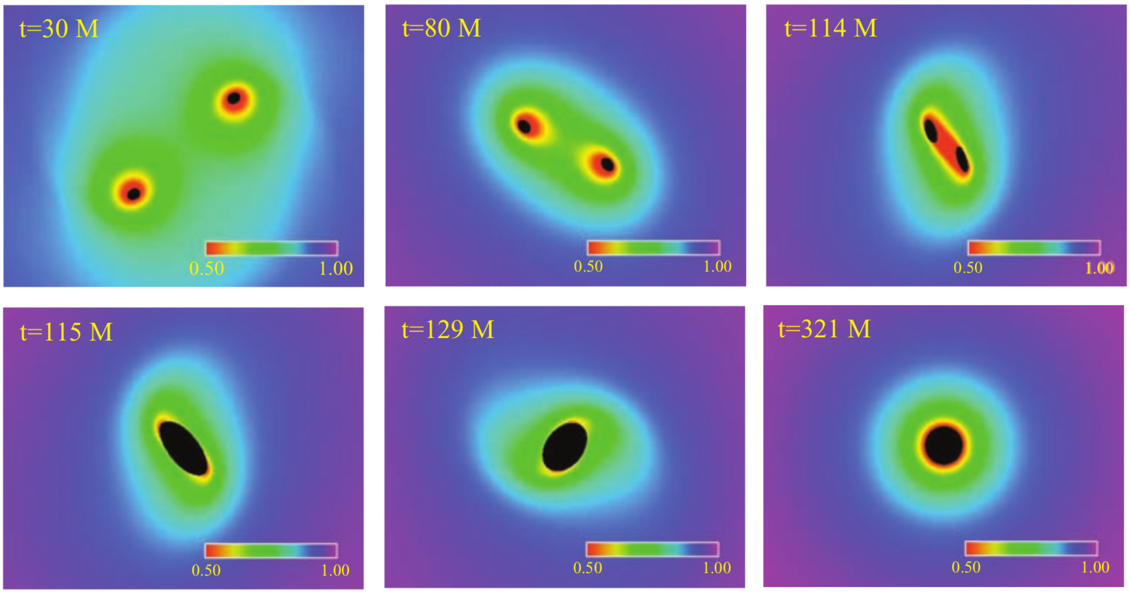

Figure 1.2: The illustration of the simulation result of Pretorius, 2005 for a binary black hole merger. The black regions are the domains excised to avoid singularities. The lapse function α, which represents the lapse speed of local proper time, is plotted as colour maps at six different timestamps. The merging black holes are equal-mass and nonspinning. The figure is cited from Baumgarte and Shapiro, 2010.

entists to implement those complex evolving systems properly in the actual code, with the support of better and faster supercomputer clusters. As a result of these breakthroughs, many successful simulations on the binary black hole merger were reported afterwards (e.g., Scheel et al., 2006; Campanelli, Lousto, and Zlochower, 2006; Baker et al., 2006).

Nowadays, there are a number of open-source numerical relativity codes available in the community, e.g., EINSTEINTOOLKIT (Löffler et al., 2012), GRCHOMBO (Clough et al., 2015), SPEC[1], and they can be used to perform high-quality simulations of merging binary black holes and provide guidance to future codes. There has been further development in the field of simulating binary black hole mergers in recent years: some investigations considered the effect of spins of the black holes on the dynamics of the systems and their gravitational wave emission (Rezzolla et al., 2008; Lousto, Healy, and Nakano, 2016), and some studies focused on simulating binary systems with eccentricity (Mroué et al., 2010; Gayathri et al., 2022). The

---

[1] `https://www.black-holes.org/code/SpEC.html`

properties of the remnants from the black hole mergers have also been a popular topic in the field (Lousto and Zlochower, 2014; Healy and Lousto, 2017).

Though it has been solved successfully, the simulation of ”standard” binary black hole mergers still has its importance as it provides a highly comprehensive benchmark of any numerical code aiming to solve Einstein equations in black hole spacetimes. Achieving stable rotating binary black holes requires proper implementation in nearly every element of a numerical relativity code, from the initial condition and gauge configuration to the evolution equations and computation grid setup. The module of gravitational wave extraction also gets examined in this benchmark to ensure its simulated signals are consistent with existing data.

In our application, we have implemented a new numerical relativity code for vacuum spacetime. By incorporating the puncture methods into it, the application also has the capability of simulating black hole systems. Compared to the existing codes, our application is based on the new technical realisation of ExaHyPE2, aiming to provide better performance in large-scale numerical relativity simulations. It is an interesting topic how new code algorithms impact the efficiency and accuracy of simulations, and the exploration becomes more crucial when dealing with non-standard black hole spacetimes like when there is modified gravity or the presence of other matter components, as their effect on the behaviour of simulations remains unknown. The evolving system we adopt is a recast version of CCZ4, which transforms the equations into a pure first-order hyperbolic formulation. This formulation is first proposed in Dumbser et al., 2018, serving as the foundation for our application. Our application has successfully conducted several numerical tests and produces preliminary results in simulating merging binary black holes, however, we are continuously working on its development to address some issues with the existing long-term instabilities. We have plans to enhance our application by adding matter components, which will allow us to perform simulations on other compact objects in the Universe, such as neutron stars. Additionally, we aim to implement more general gravity theories in our future application and investigate their effects in the context of binary black hole systems. It is worth noting that similar research on certain modified gravity theories has already appeared in recent literature (e.g., Witek et al., 2019; Okounkova et al., 2023).

1.2 ExaHyPE2: a Powerful PDE Solving Code Base

The applications we report in this thesis are based on the publicly-available code engine ExaHyPE2, which is the second generation of the ExaHyPE (An Exascale Hyperbolic PDE Engine, Reinarz et al., 2020). We name the code base "engine" as it follows the idea of separating the physics implementation and the fundamental code realisation (e.g. data structure, grid management, parallel computation), similar to a video game engine. Game developers utilise the game engine to create games according to their design, likewise, scientists can utilise the code engine to create applications according to their physics scenarios. Ideally, Research should be able to create new simulations by only specifying a "few" ingredients such as the fluxes and the code then runs out of the box. While this code design is elegant, it turns out to be quite hard to achieve as implementing state-of-the-art simulations always involves technical development as well. It is nearly inevitable to work on the technical realisations when deploying new simulations, like the ones in our applications. We therefore will cover the technical development during the implementation of our applications in Chapter 2.

In general, ExaHyPE2 is designed to simulate systems of first-order hyperbolic partial differential equations (PDEs) and relies on the Peano 4 (Weinzierl, 2019) framework for its fundamental mesh structures. The external dependencies, such as the parallelization libraries, are provided by the code segment of Technical Architecture and it is located as the bottom-most layer of our software. The layout of the whole code structure is given in figure 1.3. The core of our software is built in C++, but many tools in the repository are written in Python for readability and flexibility.

ExaHyPE2 implements a blockstructured adaptive mesh refinement (AMR) (Dubey et al., 2016) code on a spacetrees structure, which splits up the computational domain along the Peano space-filling curve (SFC) into subdomains (Li et al.,

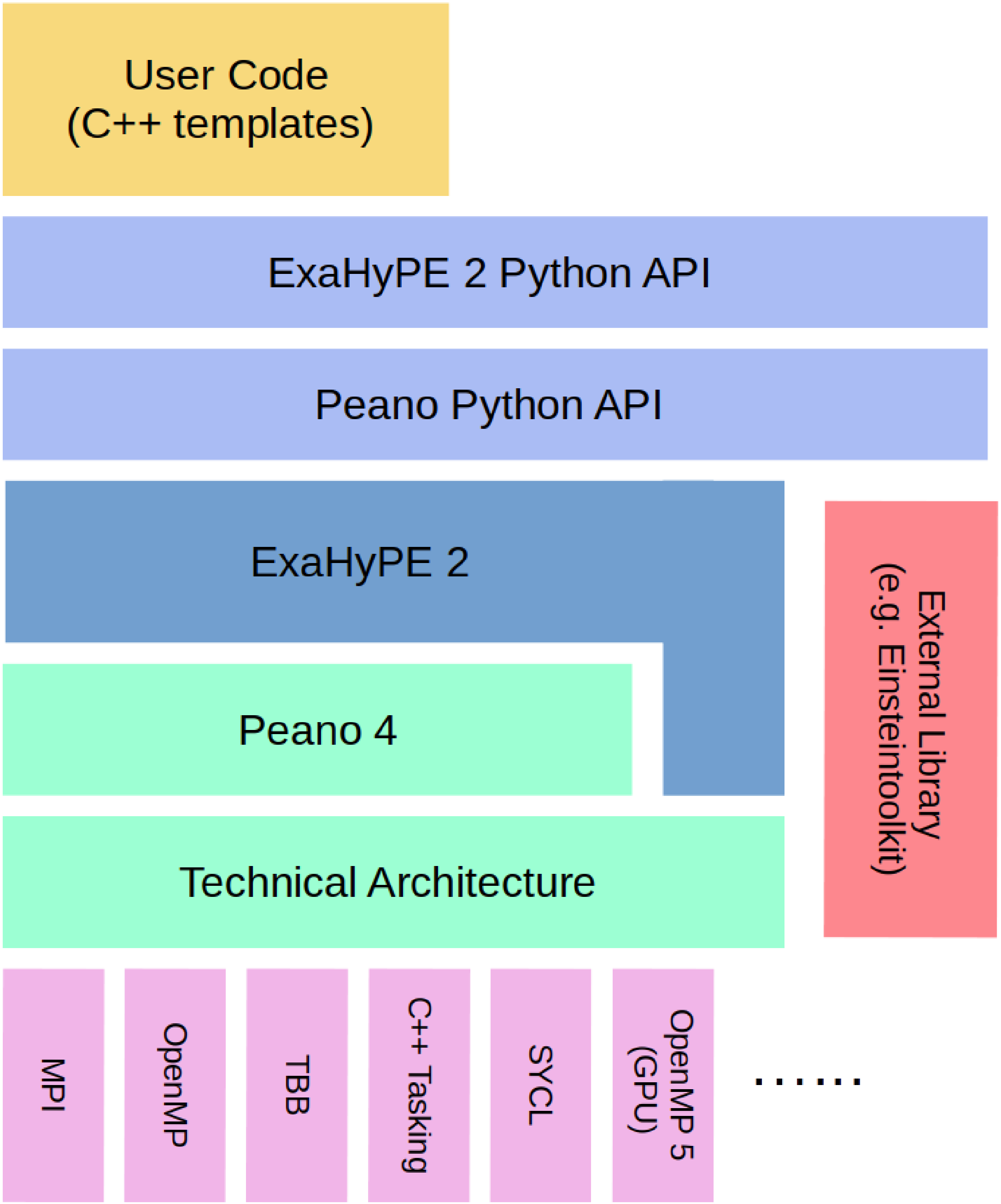

Figure 1.3: The illustration of the layout of the whole software under the name of Peano 4. Peano 4 is a framework for handling data structure in a dynamically adaptive Cartesian mesh in its fourth generation. ExaHyPE2 is a compute kernel specifically geared to solve PDEs on the meshes based on Peano 4. This thesis will only focus on the applications on the ExaHyPE2 kernel. Our code utilizes Python interfaces to generate glue codes and prepare templates for user functions. The interface is also responsible for linking other codes if needed. See Section 2.1 for more details on the structure of the Python interfaces. The code also receives external technical dependency support via the block of the Technical Architecture, which is located at the bottom-most layer of our software.

2022; Weinzierl, 2019) (see figure 1.4). All patches, which are the minimal elements during mesh refinement, are ordered along the SFC. We cut this sequence of patches into segments such that each rank gets exactly one segment hosting roughly the same number of patches. As the Peano SFC is continuous, the set of patches per rank forms a connected subdomain of the computational domain which does not overlap with any subdomain handled on another rank. Per rank, we apply the SFC splitting once more such that each thread per rank obtains its own subdomain: The patches within the computational domain are first distributed among the ranks and each rank then distributes its patches once more among the threads. This approach gives us a two-level non-overlapping MPI+OpenMP parallelization (Bungartz, Mehl, and Weinzierl, 2006; Schulz et al., 2021). The updates of individual patches per thread are mapped as a task formalism (Li et al., 2022) and we assume that the tasks can compensate for any geometric ill-balancing on the MPI level. We, therefore, do not dynamically rebalance throughout the computation.

The patches along the MPI boundaries are updated prior to other tasks such that the data transfer required for the communication of adjacent patches can overlap with further computations within each rank (Charrier, Hazelwood, and Weinzierl, 2020).

As the data structure and related technical code are handled by Peano 4 and ExaHyPE2, users are allowed to focus on the implementation of physics when building applications. This includes the implementation of the PDEs (the evolution systems), the initial and boundary conditions, eigenvalue estimations, grid configuration, as well as refinement control if an adaptive mesh is required. Users are also given the freedom to choose the computation solver in the built-in ones or define their own solver for their PDEs. Additionally, ExaHyPE2 offers interfaces where users can inject their own code segments into the source, allowing manipulations of the behaviours in different stages of the simulation. Those features give ExaHyPE2 the flexibility and versatility to address problems in different branches of sciences and engineering.

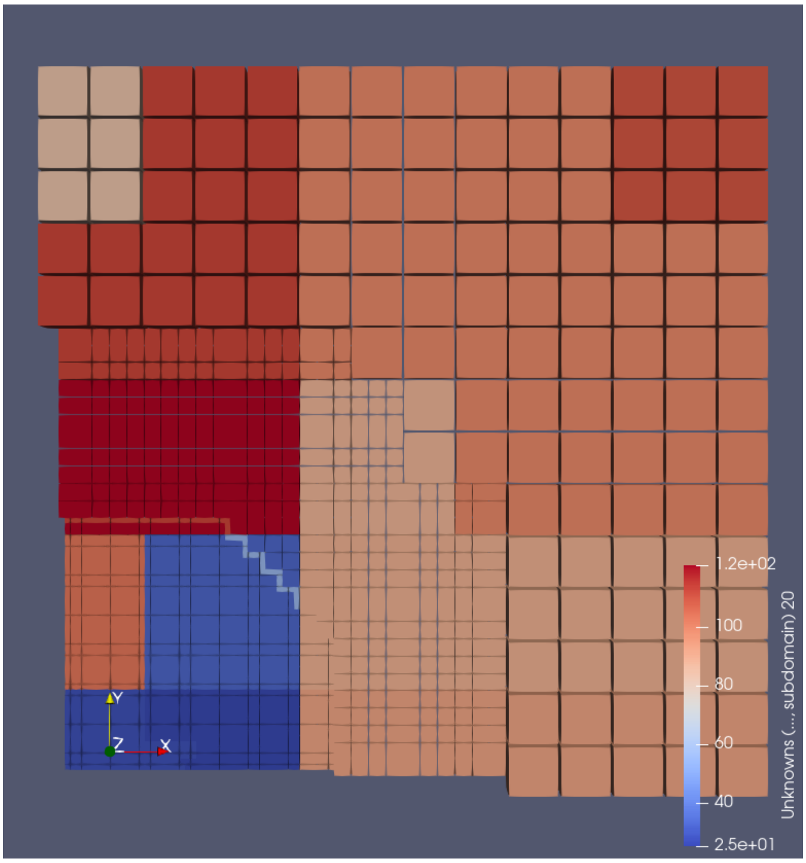

Figure 1.4: A illustration of the parallelization setup in ExaHyPE2. A two-dimensional cut through the computational mesh of a simulation as an example. The colours represent subdomains handled by different threads of different ranks. As the illustration is a cut-through, the space-filling curve structure is not visible directly.

1.3 Outline of this Thesis

In this thesis, we present the two astrophysics applications we developed on ExaHyPE2 engine, along with the technical advancement we achieved during the implementation of them. Both applications are now capable of producing promising early results for their corresponding astrophysics scenarios. However, we have also observed issues that affect the long-term stability of the simulations, and they will be discussed in the following chapters.

The rest of the content of this thesis is organized as follows: Chapter 2 presents the technical developments involved in implementing our applications. This includes the computation solvers, boundary conditions, refinement transition strategy and tracer support in ExaHyPE2. In Chapter 3, we describe our application of ExaHyPE2 designed for a spherical collapse scenario, covering the theoretical consideration, code implementation and results of simulations. We then move to introduce our second application for numerical relativity in chapter 4. This chapter provides the theoretical background of numerical relativity, details of the application structure and implementation, and reports the early simulation results of this application. Discussions are presented in both application chapters addressing existing issues and potential solutions. Finally, we summarise this thesis and give an outlook and future plan for further developing these two applications in Chapter 5.

Because the two applications are very different in scope, the notation and conventions we used may vary from chapter to chapter. To ensure consistency, we will provide the necessary notation and conventions at the beginning of each chapter if they are needed. The chapters in this thesis are written in a way that they are highly individual, allowing them to be read independently. However, we will refer to the corresponding section of the technical chapter or application chapters throughout the content, to establish a better link between the technical developments and incorporation of physics.

CHAPTER 2

Technical development on ExaHyPE2

In this chapter, we present the technical development of the ExaHyPE2 for our physics applications. We note that the fundamental framework of Peano 4 and the realization of parallel computation are outside the scope of this thesis, thus we will not cover the details of these aspects in this chapter. Readers who are interested in these topics are referred to the literature mentioned in the Introduction for more information.

This chapter begins with an introductory Section 2.1, which gives a general description of how applications are built in ExaHyPE2. It is mainly a review of the current structure of ExaHyPE2. Following that, we present the kernel computation solvers developed for our applications in section 2.2, focusing on their theoretical designs and considerations. The treatment of boundary condition is discussed in section 2.3, while the strategy of refinement transition of our code is covered in 2.4. Lastly, we give a detailed description of the tracer module in ExaHyPE2, in Section 2.5. Most of the work in the previous four sections is new and completed during the study of this PhD project. We provide an overview of those features and how they are linked to the physics scenarios of the applications in Figure 2.1.

In this chapter, we will use lower indices with Latin letters $(i, j, k, ...)$ to denote

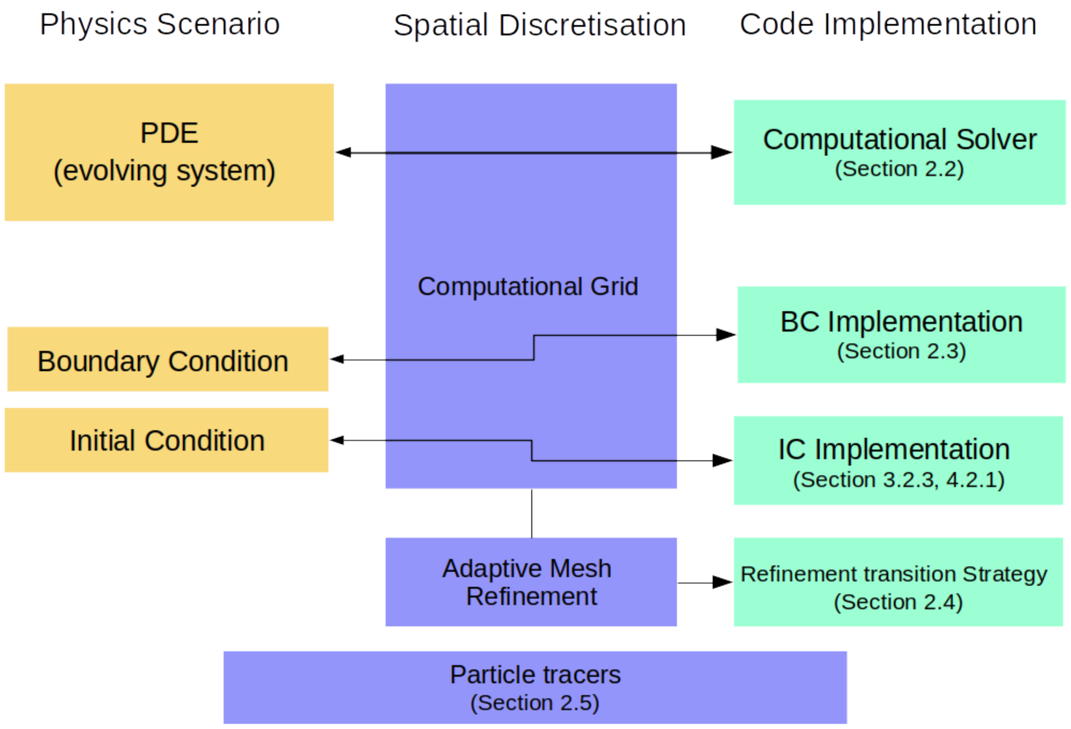

Figure 2.1: The overview of the application structure in ExaHyPE2, showing how the physics scenarios are translated into the simulation code: a physics scenario can be seen as a Cauchy initial value problem with a PDE system, an initial condition and a boundary condition. They are deployed on a computational grid following certain spatial discretisation schemes and are handled by different code modules. Our code also supports adaptive mesh refinement and has the particle tracers feature which is utilised as the data probes. We will cover those code segments in the following sections.

the spatial vectors and follow the standard Einstein convention of summing over repeated indices. The array containing evolving variables is denoted as $\vec{Q}$, where $\vec{\ }$ highlights that this is a data (rather than space) vector, e.g., in our numerical relativity application, there are 58 evolving quantities, resulting in a $\vec{Q}$ vector with a dimensionality of 58.

2.1 Building Applications in ExaHyPE2: A General Introduction

We give a brief introduction to how applications on ExaHyPE2 are constructed in this section. We first explain the spatial discretisation and define the numerical solvers, and then present how the code is built via a Python interface + template structure, under the design idea of separating the mesh traversal and the actual computation function carrying physics.

2.1.1 Spatial Discretisation and Solvers

ExaHyPE2 implements the computation grid with a block structure. The whole domain is embedded in a single cube and refined into three equal parts along each coordinate axis. This yields $3^3 = 27$ smaller cubes. We continue recursively, i.e., decide for each cube whether to refine it into 27 subcubes again, according to the refinement control function that will be introduced in Section 2.1.3. The process yields an adaptive refined Cartesian grid. Starting from this initial adaptive grid, dynamic adaptivity can be realised by the refinement and destruction of the grid cubes between time steps. We are currently not using the dynamic feature. The technical methods used to evolve the system on an adaptive grid are explained in detail in Section 2.4.

Every cube in the grid hosts a Cartesian mesh, which we name a *patch*. The patch carries the actual evolving solution and is the minimal unit of refinement. i.e. no refinement would be performed within a single patch. Every patch consists of $p \times p \times p$ mesh elements and each element holds a piecewise constant solution of the evolving system, i.e., defines one "finite volume". Every patch thus consists of p^3 volumes. The volume is the minimal unit of the computational domain and represents the finest level of discretization. Extra layers of volume are also attached at the boundary of the domain for the implementation of the boundary condition, and we call those extra layers *halos*. See Section 2.3 for more details.

To solve the system, we need to employ numerical schemes to update the solution on those patches. To be more precise, the numerical schemes should give the solution of the following PDE, which is assumed to take the first-order hyperbolic formulation:

$$\partial_t \vec{Q} + \nabla_i F_i(\vec{Q}) + B_i(\vec{Q}) \nabla_i \vec{Q} = S(\vec{Q}), \qquad \text{with } \vec{Q} : \mathbb{R}^{3+1} \mapsto \mathbb{R}^N. \tag{2.1}$$

where $\vec{Q}$ represents the array of n evolving variables. $F_i(\vec{Q})$, $B_i(\vec{Q})$ and $S(\vec{Q})$ are the flux, non-conservative product (NCP) and source term respectively. The numerical scheme provides how the solution in every volume is updated over time, thus playing a significant role in our application.

We call those numerical schemes *kernel computation solvers*. ExaHyPE2 currently supports a large variety of solvers, including different numerical methods, such as Finite Volume (FV), Finite Differences (FD), and Discontinuous Galerkin (DG), all equipped with the options of adaptive timestep and Runge-Kutta integration. The specific solvers employed in our applications are introduced in section 2.2.

2.1.2 Python Interface

The ExaHyPE2 engine provides a Python interface where users can specify the parameters, the PDE structure, and the code module they would like to employ during the simulation. The Python interface then creates templates[1] in C++ where the actual PDE and other needed functions are implemented. It also prepares and generates all necessary glue code segments and makefiles in C++ using the Jinja2 template library, then compiles the final executable file. The following list describes the

---

[1] We use the word template to refer to the empty function that users can fill later as described in Section 2.1.3. It is not related to the built-in template type in C++.

elements that are usually involved in building a typical application of ExaHyPE2:

- *Parameter Specification*: The parameters used in the application building and simulation are defined in the first part. Users can specify them in the Python interface in a hard-coded way, but one can also overwrite them through the command line argument for flexibility in code tuning. The building parameter will be used in the other parts of the code through the Python interface, such as the choice of the kernel solver and the volume size. The running parameters, on the other hand, are provided for the actual simulations, e.g., the mass of the black holes. They will be injected into the actual C++ code later through the template.

- *Kernel Solver Deployment*: As described above, a numerical solver is needed for the applications to update the solution. We select the kernel computation solver and configure it according to our needs in the Python interface. There are also several more advanced solvers under active development within the ExaHyPE2 framework, such as local time-stepping based on resolution or eigenvalues of the evolution systems, and the arbitrary-high-order-method-using-derivatives (ADER) DG method.

- *Kernel Solver Setup*: In the process of configuring the solver in the Python interface, users are required to declare the number of evolving quantities for the solver, as well as the PDE terms that they want to include in the evolution equations. This information is utilised by the Python interface to generate the template for PDE implementation accordingly. Moreover, the solver receives the parameters for grid construction, including the domain size, the resolution parameter and the strategy for refinement transitions (see section 2.4.2). They will cooperate with the AMR support of ExaHyPE2 to generate the computational grid for the simulation. The CFL ratio and the KO coefficients of the FD solver (see the section of solver below for explanations of those parameters) also get specified in the solver.

- *Code Extension implementation*: ExaHyPE2 provides the routines to inject user-defined dependency and code segments into different stages of the simulation. It is also possible for users to add auxiliary variables [2] in the code. Those auxiliary variables do not enter the evolution system, i.e., they do not require evolving equations and do not evolve in time. Instead, they can be assigned in the user-defined code which provides more output information from the simulations. The combined usage of the user-defined code and the auxiliary variables is our main approach to implementing modules for postprocessing in our code, e.g., the mass integration for the spherical accretion application (section 3.2.2) and the various modules of the numerical relativity application (section 4.2.3).

- *Tracer Configuration*: The particle support of ExaHyPE2 offers a valuable feature as the data probe in the simulation. There are two ways to utilise particles: static or moving. Each serves a distinct purpose. Static particles measure the local values of evolving quantities over time and are ideal for temporal plotting. Moving particles can be configured to follow the evolving field, essentially acting as trackers. The users need to specify the behaviours of the particles here, i.e. how the particles are initialised, how they update over time and how the associated data is outputted. The particle probe feature is discussed in detail in section 2.5.

- *Template Fill*: After the initial run of the Python interface, ExaHyPE2 will generate the templates based on the need of users. The templates contain empty functions of the user-specified PDE terms, the eigenvalue calculation, the initial and boundary conditions as well as the refinement control. Users are responsible for filling these templates according to the physics scenarios they aim to simulate. We give a further description of those templates in section 2.1.3.

- *Link external files and Compilation*: We specify the link to external code files in the final part of the application building. The external code files include the scripts where the user-defined functions are declared and defined (the ones

---

[2] Please notice the auxiliary variables here are different from the physics auxiliary variables we introduced in chapter 4 to derive our evolving system.

used in *Code Extension* above), and also the external library utilised by the application. For instance, the TwoPuncture library from EINSTEINTOOLKIT is employed in our numerical relativity code to generate initial conditions for black hole spacetime. Once all configurations and code files are in place, the Python interface of EXAHYPE2 generates the makefile. It then compiles the executable file, which we can run on a local machine, or submit to the clusters for large-scale simulation.

The list above only covered the basic elements of an EXAHYPE2 application, there are also some more advanced modules and features under testing aiming to improve the capability and flexibility of the engine. For example, it is possible to couple two solvers on the computational domain in EXAHYPE2, where the solvers evolve separately over time but also impose constraints on each other. One can use this routine to take advantage of both while avoiding their individual disadvantages, which is a potential solution to the instability issues we report in our numerical relativity application (see Section 4.3 and 4.4).

2.1.3 Application Construction through Templates

In principle, we can specify everything in the Python interface of EXAHYPE2 and there is no need to touch the real C++ file. However, it will make the Python script extremely long for some complicated applications and lose readability. Therefore, we provide users with C++ templates where one can declare and define the important code segments separately. We cover those segments below.

PDE Implementation

According to the simulated physics system, some of the terms in equation 2.1 may be absent. Users are allowed to specify the terms they need in their equation and the generated template will only ask for implementation of those terms. For instance, our spherical accretion application (see section 3.2.2) does not contain the non-conservative term, thus we only need to implement the flux and source term in the template.

The actual implementation of those terms is achieved by completing the corresponding functions in the template:

$$\begin{aligned} \text{Flux}(\vec{Q},\ x_i,\ \delta x,\ t,\ \text{d}t,\ \text{normal}) &\mapsto F_i, && (2.2)\\ \text{NCP}(\vec{Q},\ \delta\vec{Q},\ x_i,\ \delta x,\ t,\ \text{d}t,\ \text{normal}) &\mapsto B_i\delta\vec{Q}, && (2.3)\\ \text{Source}(\vec{Q},\ x_i,\ \delta x,\ t,\ \text{d}t,) &\mapsto S. && (2.4) \end{aligned}$$

Here x_i and δx are the local coordinates and volume size respectively; the latter is treated differently in various solvers, and we will discuss this in section 2.2 below. t and dt are the current time and timestep size. In most cases, the PDE terms are not dependent on the time information, and we provide these two entries here only for generalisation. Normal is the direction of the volume face[3].

Eigenvalues Estimation

Some numerical schemes in ExaHyPE2 ask users to provide an estimating routine for the eigenvalues λ of the characteristic matrices of their evolving system. These eigenvalues represent the wave-propagating speed of the quantities, and the code utilises the maximal one from all volumes and all variables to determine the timestep size if an adaptive timestep solver is employed. Certain solvers (e.g. the Rusanov finite volume solver) also require the eigenvalues in the damping term. The estimation function in the template is

$$\text{maxEigenvalue}(\vec{Q},\ x_i,\ \delta x,\ t,\ \text{d}t,\ \text{normal}) \mapsto \lambda_{\text{max}}, \tag{2.5}$$

where λ_{max} represents the maximal eigenvalue of the system. Note that here we use the word "estimation" rather than "calculation" because the eigenvalues of many evolving systems can be quite difficult, if not impossible, to calculate precisely. On the other hand, only an upper bound of the eigenvalue is needed when limiting the timestep (as shown in equation 2.17), thus users are allowed to estimate the

---

[3] Notice we always have volumes on our computational grid, however, it does not necessarily mean we are using a finite volume method. See section 2.2 below

eigenvalue roughly. However, a more precise eigenvalue can help optimize the time step size and reduce the unnecessary damping in some solvers. So we always suggest providing eigenvalues as accurately as possible.

Refinement Control

The basic parameters for resolution are already configured in the Python interface, which contains the maximal and minimal volume sizes. The ExaHyPE2 will try to refine the domain everywhere to be below the maximal volume size, which leads to a regular grid. The domain can be refined further to the intrinsic resolution limit of the minimal volume size, but the code will not do this unless the refinement control function is in place. In this function, the user can ask the code to continue refining according to certain criteria, and the implementation is provided in the template as:

$$\text{refinementCriterion}(\vec{Q},\ x_i,\ \delta x,\ t) \mapsto \text{RefinementCommand}, \tag{2.6}$$

The RefinementCommand are a set of special flag variables in ExaHyPE2 which inform the code to *keep* the current resolution or further *refine*. Users can ask the grid to further refine to a certain level, or to the limit of the minimal volume size in this function to get the specific AMR pattern they want for their simulation. ExaHyPE2 also provides the RefinementCommand of *coarse*, which plays an important role in grid destruction for dynamic AMR. As our applications only use static AMR for now, this command is not used.

Initial Condition

The function of the initial condition is provided in the templates as

$$\text{initialCondition}(x_i,\ \delta x,\ \text{gridIsConstructred}) \mapsto \vec{Q}_{\text{ini}}, \tag{2.7}$$

where $\vec{Q}_{\text{ini}}$ represents the initial array of the evolving variables. The flag gridIsConstructred is there for the purpose of optimization, as the function of the initial condition is called twice in ExaHyPE2. The first call occurs before the AMR construct, and its result is used as a reference for the refinement control function

(2.6). After the grid is constructed, The initial condition is called for the second time, in this instance, the output array represents the actual initial condition that enters the evolution later. Therefore, the quality of the result of the first call does not need to be too good. By reducing its quality, one can improve the speed of the initial condition-assignment stage of the simulation. This is especially valuable when the initial condition assignment involves expensive calculations, such as the TwoPuncture library we are using in our numerical relativity application.

Boundary Condition

Another important condition that requires implementation in the code is the boundary condition. ExaHyPE2 provides the built-in periodic boundary condition, which can be switched on directly in the Python interface. No input from users is needed in this case. For other boundary conditions, users need to complete the corresponding function in the templates:

$$\text{boundaryCondtions}(\vec{Q}_{\text{in}}, x_i,\ \delta x,\ t,\ \text{normal}) \mapsto \vec{Q}_{\text{out}}, \tag{2.8}$$

where $\vec{Q}_{\text{in}}$ and $\vec{Q}_{\text{out}}$ represent the variable arrays inside and outside the domain, and normal is the direction of the considered domain boundary. We are going to explain how the boundary condition is implemented in section 2.3.

Function Overriding

Those functions above form the essential part of the templates. Furthermore, ExaHyPE2 also allows users to override other functions in the C++ template, e.g., *startTimeStep* and *finishTimeStep*. These two functions are called at the beginning and the end of each timestep respectively, to initialise/close the current time step. This overriding feature provides users more flexibility to customize their applications according to their specific simulation requirements.

One example of this feature is in our spherical accretion application. We override the *finishTimeStep* function to reduce the mass array from every single thread accumulated to derive the global mass array for the source term (for details of this mass integration scheme, see section 3.2.2).

Another example of customization in ExaHyPE2 is overriding the solver class initializer in our numerical relativity application. In this case, the solving function from the TwoPuncture module is injected into the solver initializer. The initial condition function thus only contains the interpolation part of the puncture construction. By implementing this approach, we avoid solving the constraint equations repeatedly in the function, which reduces the computation burden of initial condition assignment significantly.

2.2 The Kernel Computation Solver

This section is dedicated to introducing the kernel computation solvers we utilised in our applications. As we mentioned above, the computation solvers are the fundamental components of the application and are responsible for providing solution update schemes for evolution. ExaHyPE2 provides various computation solvers employing different numerical algorithms. Here we only cover the two specific solvers that are implemented in our applications reported in this thesis.

2.2.1 The Rusanov Finite Volume Solver

The finite volume solver in ExaHyPE2 starts with the spatial and temporal discretization of the PDE system (2.1). We write the original PDE into the weak formulation for one timestep as:

$$\begin{aligned}\int_{\Omega\times[t,t+\delta t]} \partial_t\vec{Q}\chi \mathrm{d}x_i\mathrm{d}t = -\int_{\Omega\times[t,t+\delta t]} \left(\nabla_i F_i(\vec{Q}) + B_i(\vec{Q})\nabla_i\vec{Q}\right)\mathrm{d}x_i\mathrm{d}t \\ + \int_{\Omega\times[t,t+\delta t]} S(\vec{Q})\mathrm{d}x_i\mathrm{d}t,\end{aligned} \tag{2.9}$$

where $\mathrm{d}x_i$ runs over the simulation domain Ω, $[t, t+\delta t]$ denotes the time interval of a step and $\chi(x_i, t)$ is a test function. Equation (2.9) needs to hold for arbitrary χ to fulfil equation (2.1).

The Rusanov solver (LeVeque, 2002) we adopt in our code assumes that the solution remains constant within every timestep and every volume v, and sets all test functions as the characteristic function of one finite volume, i.e., they are $\chi_v(x_i,t) = 1$ within v and vanish anywhere else. The integration of Equation (2.9) over time gives us

$$\begin{aligned}
&\frac{1}{\delta t}\int_v \left[\vec{Q}(t+\delta t) - \vec{Q}(t)\right]\mathrm{d}x_i \\
&\qquad = -\int_v \left[\nabla_i F_i(\vec{Q}) + B_i(\vec{Q})\nabla_i\vec{Q}\right]\mathrm{d}x_i + \int_v S(\vec{Q})\mathrm{d}x_i \\
&\qquad = -\int_v \nabla_i\left[F_i(\vec{Q}) + \vec{Q}B_i(\vec{Q})\right]\mathrm{d}x_i + \int_v \vec{Q}\nabla_i B_i(\vec{Q})\mathrm{d}x_i + \int_v S(\vec{Q})\mathrm{d}x_i.
\end{aligned} \tag{2.10}$$

In the second equality, the relation $B_i(\vec{Q})\nabla_i\vec{Q} = \nabla_i[\vec{Q}B_i(\vec{Q})] - \vec{Q}\nabla_i B_i(\vec{Q})$ is used to separate the non-conservative term into a conservative part and the remainder. We then apply the divergence theorem:

$$\begin{aligned}
&\frac{1}{\delta t}\int_v \left[\vec{Q}(t+\delta t) - \vec{Q}(t)\right]\mathrm{d}x_i \\
&\qquad = -\oint_{\partial v}\left[F_i(\vec{Q}) + \vec{Q}B_i(\vec{Q})\right]\mathrm{d}S_i + \int_v \vec{Q}\nabla_i B_i(\vec{Q})\mathrm{d}x_i + \int_v S(\vec{Q})\mathrm{d}x_i,
\end{aligned} \tag{2.11}$$

where $\mathrm{d}S_i$ is the (oriented) area element of the surface of the volume v, ∂v. Here the closed-surface integration is decomposed into the summation of multiple faces that have constant normal vectors respectively. At the same time, we assume the solution vector $\vec{Q}$ to be piece-wise constant inside v, therefore we can move the $\vec{Q}$ out of the integral in the second term above:

$$\int_v \vec{Q}\nabla_i B_i(\vec{Q})\mathrm{d}x_i = \vec{Q}\int_v \nabla_i B_i(\vec{Q})\mathrm{d}x_i = \vec{Q}\oint B_i(\vec{Q})\mathrm{d}S_i. \tag{2.12}$$

and we can also apply the following replacement:

$$\int_v \bigcirc\mathrm{d}x_i \to \bigcirc V_v, \quad \oint_{\partial v}\bigcirc_i\mathrm{d}S_i \to \sum_{\partial v}\bigcirc_i n_i S_{\partial v}, \tag{2.13}$$

where V_v is the volume of v; $S_{\partial v}, n_i$ are the area and unit normal vector of one face of ∂v, respectively; and $\bigcirc$ ($\bigcirc_i$) denotes a generic scalar (space vector) function. This leads to the final explicit Euler time stepping scheme we implemented in the code:

$$\vec{Q}(t+\delta_t) = \vec{Q}(t) + \delta t S(\vec{Q}) + \sum_{\partial v} \mathrm{Flux}^{\pm}(\vec{Q})\Big|_{\partial v}, \tag{2.14}$$

The indices on the flux terms indicate the relative position of the face and the considered volume. As our grid utilises a cubic (or square, on a two-dimensional grid) volume, there are three (two) pairs of Flux$^{\pm}$, where $-$ means the face is on the left of the volume while $+$ means the face is on the right of the volume. Assuming the variable arrays of the left, considered and right volumes are $\vec{Q}^{-}$, $\vec{Q}$, and $\vec{Q}^{+}$ respectively (see figure 2.2), we shall have

$$\begin{aligned}\mathrm{Flux}_d^{-} = \frac{\delta t}{\delta x}\Bigg[&\frac{1}{2}\big(F_d(\vec{Q}) + F_d(\vec{Q}^{-})\big) - \frac{1}{2}\left(\vec{Q} - \vec{Q}^{-}\right) B_d\left(\frac{1}{2}\left(\vec{Q} + \vec{Q}^{-}\right)\right) \\ &- \eta \max\left(\lambda_{\max}(\vec{Q}), \lambda_{\max}(\vec{Q}^{-})\right)\left(\vec{Q} - \vec{Q}^{-}\right)\Bigg],\end{aligned} \tag{2.15}$$

$$\begin{aligned}\mathrm{Flux}_d^{+} = \frac{\delta t}{\delta x}\Bigg[&\frac{1}{2}\big(F_d(\vec{Q}^{+}) + F_d(\vec{Q})\big) - \frac{1}{2}\left(\vec{Q}^{+} - \vec{Q}\right) B_d\left(\frac{1}{2}\left(\vec{Q}^{+} + \vec{Q}\right)\right) \\ &+ \eta \max\left(\lambda_{\max}(\vec{Q}^{+}), \lambda_{\max}(\vec{Q})\right)\left(\vec{Q}^{+} - \vec{Q}\right)\Bigg],\end{aligned} \tag{2.16}$$

where the lower index d represents the normal direction and takes the value of $\{1,2,3\}$ ($\{1,2\}$ in 2D). The contribution of flux term F_d on the faces is approximated by taking the averages of the flux in the two adjacent volumes, in which we have assumed the flux term is linear on $\vec{Q}$. On the other hand, the contribution from the non-conservative product term uses the average of the variable array itself as it is not linear most of the time. The result then is corrected (limited) with a term of numerical diffusion (LeVeque, 2002), which is proportional to the largest eigenvalue $\lambda_{\max}$ of the characteristic matrix. η is a problem-specific constant and has a default value of 0.5.

To achieve a stable evolution, the timestep of our finite volume solver δt is subject to the Courant–Friedrichs–Lewy (CFL) condition with

$$\delta t < C\frac{|\delta v|}{\lambda_{\max}}, \tag{2.17}$$

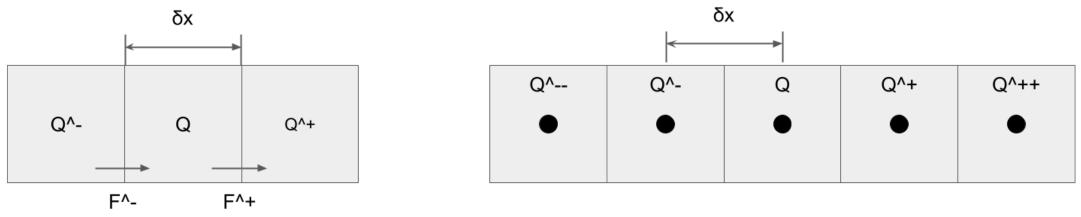

Figure 2.2: The stencil in one dimensional of the two solvers we describe in this section. They shared the same grid structure fundamentally (volumes). *Left panel*: The stencil of the Rusanov finite volume solver, where the flux from the left and right of the considered volume are labelled accordingly. The variable array in the volume is assumed to be piece-wise constant. *Right panel*: The five-point stencil for the fourth-order finite difference solver. The volume center is used as the grid point as marked.

where $C < 1$ is a user-specified safety parameter that may vary in different simulations. Our scheme employs a global time-stepping scheme and thus uses the smallest global volume length $|\delta x|$. It remains invariant over time as we fix the finest resolution in our simulations. The maximum eigenvalue $\lambda_{\max}$, however, changes over time and thus has to be recalculated after each step. Our timestep size would then adapt accordingly.

We employ this Rusanov Finite Volume solver in our spherical collapse application, and it yields quite satisfying outcomes in reconstructing self-similar behaviours of this scenario, both in standard and modified gravity. Furthermore, it also shows a good property in capturing and resolving shocks in the computational domain. For more details on the performance of this solver in real physics simulations, we direct the readers to section 3.3 in the application chapter.

2.2.2 The 4th Order Finite Difference Solver

As we mentioned above, we can construct the finite difference scheme in ExaHyPE2, while having the basic structure of volumes. By assuming the solution to be piece-wise constant in the volume, the variable array of a volume is identical to the array located at the center of this volume. This insight allows us to construct a grid of discrete points, where the centers of every volume are treated as points and carry the array of evolving variables. In the remainder of this subsection, we will use the word grid points rather than volume centres to be consistent with the scheme design.

To approximate the derivatives in fourth-order accuracy, we utilise a five-point stencil in one dimension as shown in figure 2.2:

$$\left.\frac{\partial Q\left(x\right)}{\partial x}\right|_{x_0} = \frac{1}{12\delta x}\left[8\left(Q^{+} - Q^{-}\right) - \left(Q^{++} - Q^{--}\right)\right] + \mathcal{O}\left(\delta x^4\right). \tag{2.18}$$

Where x_0 is the considered grid point, the $+$ and $-$ upper indices of quantities indicate the relative position of the indexed quantities with the current grid point, and the number of signs specifies the distance, e.g., $Q^{++} := Q(x_0 + 2\delta x)$, $Q^{-} := Q(x_0 - \delta x)$. We have omitted the vector sign of the evolving variables Q for simplicity. This stencil achieves fourth-order accuracy which can be seen from the following Taylor expansions:

$$\begin{aligned} Q^{+} &= Q\left(x_0 + \delta x\right) && (2.19)\\ &= Q\left(x_0\right) + \frac{\partial Q}{\partial x}\left(x_0\right)\delta x + \frac{1}{2}\frac{\partial^2 Q}{\partial x^2}\left(x_0\right)\delta x^2 + \frac{1}{6}\frac{\partial^3 Q}{\partial x^3}\left(x_0\right)\delta x^3 + \mathcal{O}\left(\delta x^4\right), \\ Q^{-} &= Q\left(x_0 - \delta x\right) && (2.20)\\ &= Q\left(x_0\right) - \frac{\partial Q}{\partial x}\left(x_0\right)\delta x + \frac{1}{2}\frac{\partial^2 Q}{\partial x^2}\left(x_0\right)\delta x^2 - \frac{1}{6}\frac{\partial^3 Q}{\partial x^3}\left(x_0\right)\delta x^3 + \mathcal{O}\left(\delta x^4\right), \\ Q^{++} &= Q\left(x_0 + 2\delta x\right) && (2.21)\\ &= Q\left(x_0\right) + 2\frac{\partial Q}{\partial x}\left(x_0\right)\delta x + 2\frac{\partial^2 Q}{\partial x^2}\left(x_0\right)\delta x^2 + \frac{4}{3}\frac{\partial^3 Q}{\partial x^3}\left(x_0\right)\delta x^3 + \mathcal{O}\left(\delta x^4\right), \\ Q^{--} &= Q\left(x_0 - 2\delta x\right) && (2.22)\\ &= Q\left(x_0\right) - 2\frac{\partial Q}{\partial x}\left(x_0\right)\delta x + 2\frac{\partial^2 Q}{\partial x^2}\left(x_0\right)\delta x^2 - \frac{4}{3}\frac{\partial^3 Q}{\partial x^3}\left(x_0\right)\delta x^3 + \mathcal{O}\left(\delta x^4\right), \end{aligned}$$

The stencil eliminates the $\delta x^0, \delta x^2$ and δx^3 terms so that only terms linear in δx remain. Notice that we can apply this formulation to any spatial-dependent function, therefore the divergence of the flux is given as:

$$\left.\frac{\partial F}{\partial x}\right|_{x_0} = \frac{1}{12\delta x}\left[8\left(F^{+} - F^{-}\right) - \left(F^{++} - F^{--}\right)\right] + \mathcal{O}\left(\delta x^4\right). \tag{2.23}$$

Equipped with the relations (2.18) and (2.23), we can derive the time stepping scheme of this fourth-order finite difference (FD4) solver

$$\begin{aligned}\vec{Q}(t+\delta t) = \vec{Q}(t) + \delta t S(\vec{Q}) - \sum_i \frac{\delta t}{12\delta x_i}\left[8\left(F_i^{i+} - F_i^{i-}\right) - \left(F_i^{i++} - F_i^{i--}\right)\right] \\ - \sum_i B_i \frac{\delta t}{12\delta x_i}\left[8\left(Q^{i+} - Q^{i-}\right) - \left(Q^{i++} - Q^{i--}\right)\right],\end{aligned} \tag{2.24}$$

where we have used the forward finite differences for the time derivatives $\partial_t \vec{Q} = \left(\vec{Q}(t+\delta t) - \vec{Q}(t)\right)/\delta t$ for a timestep size of δt. The index i goes from 1 to 3 and indicates the direction of the neighbouring points we are looking into. The five-point stencil of (2.18) is the central difference and it is our preferred choice in most cases. However, some terms may need a lopsided stencil to capture specific features or behaviours, especially those that may lead to asymmetric patterns in the system, e.g. the advection terms of the CCZ4 system (Radia et al., 2022) in numerical relativity, which appears in our application as well. The left and right lopsided stencils with also the fourth-order accuracy are

$$\left.\frac{\partial Q(x)}{\partial x}\right|_{x_0} = \frac{1}{12\delta x}\left[-Q^{---} + 6Q^{--} - 18Q^{-} + 10Q + 3Q^{+}\right] + \mathcal{O}\left(\delta x^4\right), \tag{2.25}$$

$$\left.\frac{\partial Q(x)}{\partial x}\right|_{x_0} = \frac{1}{12\delta x}\left[-3Q^{-} - 10Q + 18Q^{+} - 6Q^{++} + Q^{+++}\right] + \mathcal{O}\left(\delta x^4\right), \tag{2.26}$$

which we shall switch to accordingly when needed.

The timestep size of the FD4 solver is also subject to the CFL condition (2.17) above. In practice, we find that a smaller safe parameter C is usually needed for the FD4 solver, compared to the case of the finite volume solver, to achieve a stable evolution of the system.

Another essential piece of the FD4 solver is numerical dissipation. The numerical errors that appear in the finite difference scheme usually have a wavelength of the grid size δx, and such spurious high-frequency modes have a faster growth rate compared to the actual PDE solution, which is low-frequency. As a result, the accumulation and amplification of these numerical errors are very likely to disrupt the evolution and lead to a code crash. This is why we need the numerical dissipation just like the one introduced in the FV solver above. The dissipation term in the

FD4 solver is introduced as a low-pass filter that helps suppress the high-frequency modes in the solution, while only introducing high-order errors in the actual solution. This guarantees the dissipation terms approach zero when the continuous limit δx is taken. In our solver, we choose the standard Kreiss-Oliger (KO) dissipation at order $N = 3$, which has a form in one dimension as(Kreiss and Oliger, 1973):

$$\mathrm{KO}^{(3)} = \frac{\epsilon}{64\delta x}\left(Q^{---} - 6Q^{--} + 15Q^{-} - 20Q + 15Q^{+} - 6Q^{++} + Q^{+++}\right), \quad (2.27)$$

where ϵ is a user-defined parameter to control the strength of the numerical dissipation. By checking the Taylor expansion of every term above, one can see that

$$\mathrm{KO}^{(3)} = \frac{\epsilon}{64}\frac{\partial^6 Q}{\partial x^6}\delta x^5 + \mathcal{O}\left(\delta x^6\right). \quad (2.28)$$

Therefore the new numerical error introduced by this KO term is one order higher than that of the solver itself. The final numerical dissipation term is the sum over all directions of (2.27).

We develop this fourth-order finite difference solver for our numerical relativity application since the tests show that the finite volume solver is too dissipated for black hole spacetimes. It shows a good capability to resolve the puncture structure of the black holes and gives a rather stable evolution of the dynamic black hole systems. However, it has been noted that there are instabilities presented in simulations of static single black holes using this FD4 solver. This is currently being investigated. For the report of the performance and discussions of existing issues, we refer readers to sections 4.3 and 4.4 in the application chapter accordingly.

Before we close this section, it should be pointed out that the FV and FD solvers we describe here both have the capability to integrate using the high-order Runge-Kutta scheme, which can provide enhanced temporal accuracy in simulations. However, in our applications, we have chosen to utilise the Euler integration scheme as the Runge-Kutta scheme does not improve the accuracy significantly. At the same time, the Euler scheme also avoids the high computational overhead caused by the Runge-Kutta scheme.

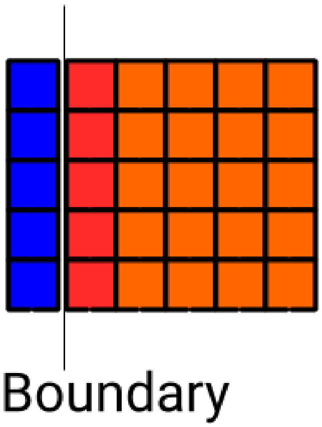

Figure 2.3: An illustration of the boundary layout used in our simulations with ExaHyPE2, when we use the Finite Volume kernel. Outside the boundary of the simulation domain, denoted by the thin black line, a layer of ghost volumes (blue squares) are set up, and the interested quantities in the ghost volumes, $\vec{Q}_{\text{out}}$, is determined by the values of these quantities, as well as their derivatives, in the volumes immediately inside the boundary (dark red squares), $\vec{Q}_{\text{in}}$. The figure is cited from the document of code repository `https://gitlab.lrz.de/hpcsoftware/Peano`.

2.3 Boundary Condition Implementation

The ideal "boundary condition" is a domain with an infinite size as in the real Universe. Clearly, it is not possible in numerical simulations with finite computation resources. Therefore, we need to specify the behaviour of our solution at the domain boundary, i.e., how to perform the time stepping (2.14) or (2.24) when the neighbours on the certain side are out of the boundary.

ExaHyPE2 utilises the halos of ghost volumes of the domain to do this job. Those halos are artificial layers of volumes that can be considered as an extension of the actual physical domain. The number of halo layers depends on the solvers we employ. When the volumes at the edge of the physical domain perform their time stepping, they use the values in those ghost volumes as the neighbouring values. Those ghost volumes do not evolve themselves but instead receive their values from the boundary condition function, which we introduced in Section 2.1.3.

Figure 2.3 illustrates the layout of volumes at the boundary for the finite volume solver. As the time-stepping scheme of the FD solver 2.14 only requires its direct neighbour, only a single halo layer of ghost volumes is needed (blue squares in the figure). The boundary function (2.8) specifies how the values in those blue squares, $\vec{Q}_{\text{out}}$, depend on their direct neighbours (dark red squares), $\vec{Q}_{\text{in}}$. While (2.8) only provides the entries for the variable array $\vec{Q}_{\text{in}}$ itself, it is also possible to add the dependence from the derivatives of quantities in those dark red volumes. This is achieved by utilising the code extension feature we introduced in section 2.1: we calculate the derivatives of the evolving quantities and attach them within the variable array $\vec{Q}$ as auxiliary variables. Consequently, the array $\vec{Q}_{\text{in}}$ that the boundary condition receives would contain the information of derivatives. This approach plays an important role in implementing the hybrid boundary condition in the spherical collapse application (see section 3.2.3).

On the other hand, the boundary layout for the FD4 solver of ExaHyPE2 is slightly more complicated as the time-stepping (2.24) and dissipation term (2.27) of it asks for neighbours two/three points (volumes) away. Therefore we need three halo layers of ghost volumes (see figure 2.4) for the FD4 solver. To assign the boundary condition for multiple layers, we adopt the 1 → 1 scheme to specify the values in the ghost volumes layer by layer. i.e., we first assign the values $Q_{3,j,k}$ use $Q_{2,j,k}$ by treating them as Q_{out} and Q_{in} in boundary condition function, then assign the values $Q_{4,j,k}$ use $Q_{3,j,k}$, assign the values $Q_{4,j,k}$ use $Q_{3,j,k}$. All three layers are now filled. Intuitively speaking, the boundary condition of the FD4 solver is achieved by applying the boundary condition of the FV solver three times.

2.3.1 Simple Boundary Conditions

According to the formulation of the boundary condition function (2.8), it is straightforward to implement the Dirichlet-type conditions:

$$\vec{Q}_{\text{out}} = f(x_i,\ \delta x,\ t,\ \text{normal}), \tag{2.29}$$

and the Neumann-type conditions

$$\frac{\vec{Q}_{\text{out}} - \vec{Q}_{\text{in}}}{\delta x_{\text{normal}}} = f(x_i,\ \delta x,\ t) \ \rightarrow\ \vec{Q}_{\text{out}} = \vec{Q}_{\text{in}} + \delta x_{\text{normal}} f(x_i,\ \delta x,\ t), \tag{2.30}$$

in ExaHyPE2. We can further combine these two types of conditions to get the Robin(Fourier)-type boundary conditions, which can also be seen as the mix of the two boundary conditions above:

$$\begin{aligned} C_0\vec{Q}_{\text{out}} + C_1\frac{\vec{Q}_{\text{out}} - \vec{Q}_{\text{in}}}{\delta x_{\text{normal}}} &= f(x_i,\ \delta x,\ t) \\ &\rightarrow \\ \vec{Q}_{\text{out}} &= \left(C_0 + \frac{C_1}{\delta x_{\text{normal}}}\right)^{-1}\left(f(x_i,\ \delta x,\ t) + \frac{C_1}{\delta x_{\text{normal}}}\vec{Q}_{\text{in}}\right). \end{aligned} \tag{2.31}$$

We have used x_{normal} to represent the volume length (grid size) in the normal direction specified in the boundary condition. C_0 and C_1 are the parameters that are specified according to the boundary condition needed. As we introduced in section 2.1.3, ExaHyPE2 also provides the newly-developed built-in periodic boundary condition. It is very useful in wave propagation tests, such as the gauge wave test of our numerical relativity application (see section 4.3.1).

The boundary condition function can also receive information about the derivatives of the $\vec{Q}_{\text{in}}$ using the feature of auxiliary variables as described above. In this case, the derivative of the quantities is actually evaluated by accessing an extra layer into the physics domain, i.e., the orange squares in the column right next to the dark red ones in figure 2.3. The derivatives are calculated as the second-order finite difference scheme:

$$\partial_{\text{normal}}\vec{Q}_{\text{dark red}} = \frac{\vec{Q}_{\text{right}} - \vec{Q}_{\text{left}}}{2\delta x_{\text{normal}}}, \tag{2.32}$$

where $\{\text{left}, \text{right}\} \in \{\text{blue}, \text{orange}\}$ depending on the orientation of the considered boundary.

Those simple boundary conditions are generally sufficient for various applications in ExaHyPE2, however, they behave badly in scenarios inhabiting outgoing waves, such as the case of rotating binary black holes. We, therefore, need more advanced boundary conditions for those physics processes.

2.3.2 Sommerfeld Radiation Condition

The default boundary condition of ExaHyPE2 application is the homogeneous Newmann condition which asks the derivatives at the boundary to vanish:

$$\frac{Q_{\text{out}} - Q_{\text{in}}}{\delta x_{\text{normal}}} = 0 \ \rightarrow \ Q_{\text{out}} = Q_{\text{in}}, \tag{2.33}$$

In our numerical relativity application, this naive outflow boundary condition turns out to be insufficient for black hole scenarios because it resolves the solution inaccurately, thus leading to a wave-like reflection.

To improve this, we now consider the Sommerfeld (radiative) boundary condition. It is one kind of absorbing boundary condition and it enforces that the wave at the boundary only travels out of the domain. Its basic formulation can be written as

$$\partial_t Q + c\partial_{\text{normal}} Q = 0\Big|_{\partial\Omega}. \tag{2.34}$$

Here c is the wave speed, i.e., the eigenvalue of the characteristic matrix of the evolution system. We can apply this boundary condition directly if our wave is travelling perpendicular to the boundary, but that is not true in our case as we have a spherical symmetric system in a cubic domain. If we are far enough from the centre region, we should expect the solution in the form of $u(r - ct)/r$ for a spherical symmetry setup, where r is the radius from the system origin. It leads to the Sommerfeld boundary condition as

$$\partial_t Q + c\partial_r Q + cQ/r = 0\Big|_{\partial\Omega}. \tag{2.35}$$

Now we need to project the radical derivative to its Cartesian component. As we have a spherically symmetric function, the following relation,

$$\partial_i Q = \frac{\partial r}{\partial x_i}\partial_r Q = \frac{x_i}{r}\partial_r Q, \tag{2.36}$$

can be used to give

$$\partial_t Q + c\frac{r}{x_i}\partial_i Q = -\frac{c}{r}(Q - Q_{\mathrm{ini}})\Big|_{\partial\Omega}, \quad i \in \{1,2,3\} \tag{2.37}$$

Here Q_{ini} is the value of evolving quantities at the boundary in the initial condition, as it serves as a "background" and thus does not enter the wave-like equation.

We can also use another formulation from the inverse relation of (2.36):

$$\partial_r Q = \sum_i \frac{\partial x_i}{\partial r}\partial_i Q = \sum_i \frac{x_i}{r}\partial_i Q \tag{2.38}$$

It gives another version of the Sommerfeld boundary condition:

$$\partial_t Q + c\sum_i \frac{x_i}{r}\partial_i Q = -\frac{c}{r}(Q - Q_{\mathrm{ini}})\Big|_{\partial\Omega} \tag{2.39}$$

The sum-up notation is added here deliberately to distinguish from the first approach where double i on the left is not a sum-up in (2.37).

In the continuous limit, these two versions should give the same results. In numerical simulations, however, they are different and both have advantages and disadvantages. The first version (2.37) is much easier to implement but leads to potential numerical error in systems where the exact spherical symmetry is lost (like the case of the binary black hole spacetime), as the relation 2.36 does not hold strictly anymore. The second version (2.39), on the other hand, behaves better in these situations, but it requires derivatives parallel to the boundary. This makes the boundary condition more complicated and is also troublesome when we hit the edge of the boundary where we lose access to the volume on one side. Currently, the first version of the Sommerfeld boundary condition is employed in the code.

We then explain the implementation of the Sommerfeld boundary condition in code practice, with three halo layers of ghost volumes. Without loss of generality, we assume the considered boundary is located in the positive x position. Figure 2.4 illustrates the boundary layout in this case, where the light red squares on the left are the volumes inside the domain while the light blue ones on the right are the ghost volumes that we need to assign in the boundary condition. The six layers are

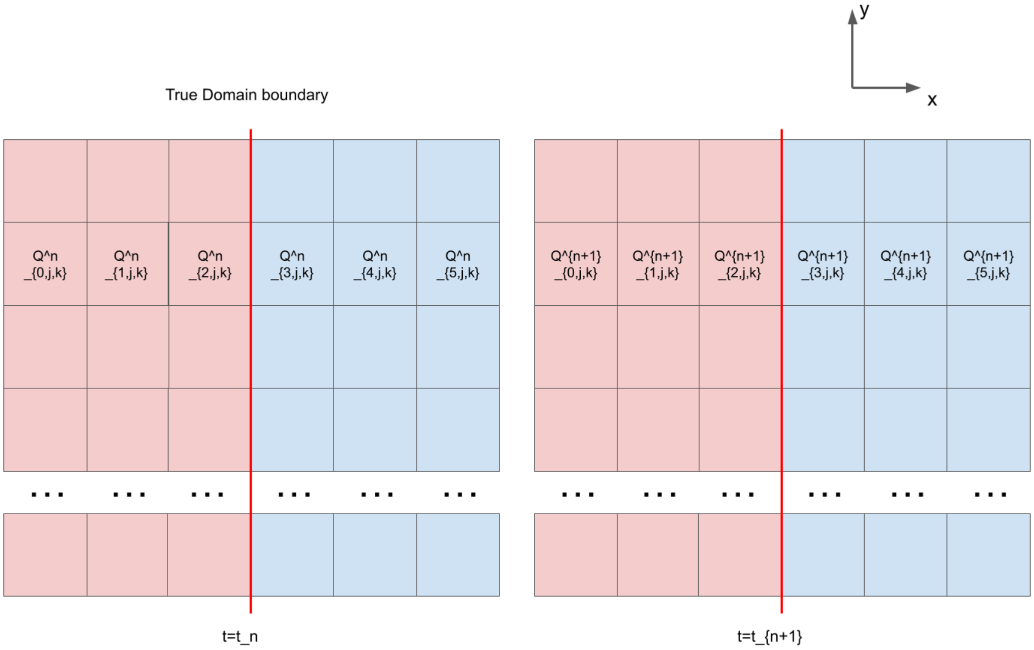

Figure 2.4: (One dimension is omitted) Illustration of the entries we need for implementing the Sommerfeld condition for three halo layer setup. The boundary on the positive x is shown here, thus the right half (light blue squares) is outside the domain. We need to fill the outside layers of the boundary ($Q^{n+1}_{3,j,k}$, $Q^{n+1}_{4,j,k}$, $Q^{n+1}_{5,j,k}$) from values inside the domain (light red squares, $Q^{n+1}_{0,j,k}$, $Q^{n+1}_{1,j,k}$, $Q^{n+1}_{2,j,k}$), and the information from previous timestep ($Q^{n}_{\{0-5\},j,k}$). See the text content for more details.

labelled from 0 to 5 from left to right as their lower indices in the x position.

As we describe above, we can implement the boundary condition in a $1 \to 1$ manner, first considering the volumes at layers 2 and 3. The Sommerfeld boundary condition (2.37) gives

$$\frac{1}{\delta t}\left(Q_3^{n+1} - Q_3^n\right) + c\frac{r}{x_i \delta x_i}\left(Q_3^n - Q_2^n\right) = -\frac{c}{r}\left(Q_3^n - Q_{\text{ini}}\right). \tag{2.40}$$

The indices for the y and z directions are omitted as they are all the same in this case. The upper indices n and $n+1$ represent the previous and current timestep: One important feature of the Sommerfeld boundary condition is that it requires information from the previous timestep to evaluate the time derivatives. The variables from the previous timestep are not available in boundary condition function (2.8) by default, and access to them once again utilises the feature of code extension we introduced in section 2.1.

Using the expression (2.40), we can derive the values of variables in layer 3 and then we apply the same manipulation for layer 3 to layer 4, layer 4 to layer 5 to fill all three halo layers. Notice the r, c and x_i in the expression need to change according to the considered layers.

We can "upgrade" the expression above to enhance the accuracy of the Sommerfeld condition by taking the averages of two layers for the quantities, following the idea of Alcubierre, 2008:

$$\begin{aligned}\frac{1}{\delta t}\left[\left(Q_3^{n+1} - Q_3^n\right) + \left(Q_2^{n+1} - Q_2^n\right)\right] + c\frac{r}{x_i \delta x_i}\left[\left(Q_3^{n+1} - Q_2^{n+1}\right) + \left(Q_3^n - Q_2^n\right)\right]& \\ = -\frac{c}{r}\left[\left(Q_3^n + Q_2^n\right) - 2Q_{\text{ini}}\right].&\end{aligned} \tag{2.41}$$

This equation also gives Q_3^{n+1}.

For the second Sommerfeld BC approach, we replace the second term in (2.40) above with:

$$\frac{c}{r}\left[\frac{x}{\delta x}\left(Q_{3,j,k}^n - Q_{2,j,k}^n\right) + \frac{y}{2\delta y}\left(Q_{3,j+1,k}^n - Q_{3,j-1,k}^n\right) + \frac{z}{2\delta z}\left(Q_{3,j,k+1}^n - Q_{3,j,k-1}^n\right)\right] \tag{2.42}$$

Similarly, it can be upgraded to a more accurate version by averaging with the corresponding difference at timestep $n+1$. As we know everything at the boundary at the timestep n, it is straightforward to implement. Just notice we need to change the corresponding central difference to the lopsided difference when we hit the edge of the boundary. This approach is not implemented in the current code.

The Sommerfeld boundary condition implemented as described above shows a significant numerical improvement compared to the homogeneous Neumann condition, especially in the simulations of black hole spacetimes. It provides a much more stable boundary behaviour and suppresses the reflection which ruins the central solution. However, it can not eliminate the boundary effect completely, and we observe that long-term simulations are still suffering from errors propagating inward from the boundary. As the Sommerfeld condition performs optimally in systems with spherical symmetry, one potential solution to address this issue is to increase the domain size. We are actively looking for a resolution for the boundary effect in ongoing research.

2.4 Patch Communication and Adaptive Mesh Refinement

In this section, we present how we achieve solution updates on the adaptive mesh in ExaHyPE2 via the halo-like structures called "face". The face structures are introduced as the bridge among patches as they may be handled by different threads in a parallel computation setup.

2.4.1 Patch Communication through Face Structures

In parallel computers, the patch is the minimal domain unit distributed among cores, thus cores may have no access to the neighbours for those volumes located at the edges of patches, as these neighbours belong to the patch processed by other cores when we use a non-overlapping domain decomposition. As the communication among different threads is rather expensive, we thus introduce the halo of the patches to achieve an overlapping domain decomposition. The halo can be considered as the

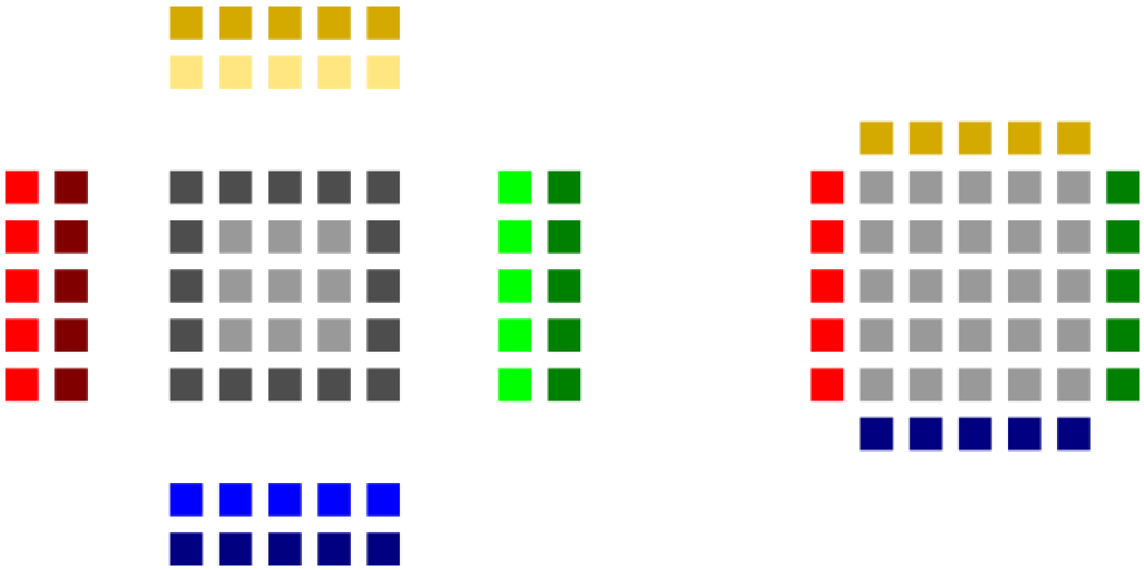

Figure 2.5: The illustration of patches and faces structure on the computation domain in ExaHyPE2. A two-dimensional patch with 5 × 5 mesh elements (volumes) and a halo of one layer is given. On the right panel, we show the structure of this patch during the time-stepping: the patch (grey squares) has a halo (coloured squares) that is used to update the volumes at its edges. The values in the halos are from the face structures (coloured squares) we showed on the left. The face structures do not enter the actual evolution and are only responsible for bridging different patches. The faces are the exact copies of the corresponding part of adjacent patches, e.g., the halves that are close to this patch are the copies of those boundary volumes of it(dark grey squares). See the text content for more details. The figure is cited from the document of code repository https://gitlab.lrz.de/hpcsoftware/Peano.

local boundary condition: the volumes on the edges of the patches utilised the information in the halo layers to perform time stepping. Clearly, as mentioned above, the number of the halo layers of the patches also depends on the solver we employ.

The right part of Figure 2.5 gives an example of the patch and halo structures. We assume a two-dimensional patch with 5 × 5 volumes and a halo of one layer in this instance. During the evolution, the core performs the timestepping within the patch (grey squares) and utilises the halo (coloured squares) to update the volumes on the edge of the patch. As we are now inside the simulation domain, the halos do not need a user function to assign like what is needed for the boundary conditions; instead, they are determined by the boundary volumes of neighbouring patches. This is achieved by introducing the above-mentioned “face” structure in our code, which is responsible for performing communication among patches. Every face structure holds $2k \times p$ volumes, assuming the halo layer has a thickness of k and the patch size p. So in our example above, each face has 2×5 volumes and is shown as the coloured squares on the left in the figure. The values of the volumes in the face structures are the exact copies of the boundary part of the adjacent patches. For instance, the two red columns on the left represent a face that bridges the patch shown in the figure and its left neighbour patch. Those volumes marked in dark red are copies of the boundary part of the current patch (leftmost column in dark grey), while the volumes in light red are copies of the volumes in the rightmost column of the neighbour patch. The case with multiple halo layers is similar and we just need to copy more columns from both patches to the face. Intuitively speaking, the patches make up a non-overlapping domain decomposition of the computational domain, while the faces introduce an overlap among them.

The copy from patches to faces takes place after every timestep and we assign the halo layers of every patch utilising the face at the beginning of the next timestep. That is how the halo layer of patches receives information from their neighbours. No direct data communication would happen among patches in ExaHyPE2 and faces take all the responsibility for that and themselves do not enter the actual evolution. Clearly, those patches that happen to be located at the boundary of the physical domain have no neighbours for their halos, and they thus need to be assigned using the boundary function we describe above in section 2.3.

2.4.2 Trilinear Interpolation in the Tensor Product Formalism

The copy back from the face structure to the halo layers of patches is relatively straightforward in a regular grid as it is a direct one-to-one map, such as from the five light red squares on the left of the figure 2.5 to the five light red squares on the right. However, special attention needs to be paid when dealing with the boundary of the refinement transition.

We use the word refinement transition to indicate the resolution change between patches. As we mentioned above, the adaptive mesh in ExaHyPE2 is constructed by refining patches recursively in three partitions and creating new patches accordingly. There is no refinement occurring within the patches as the patches are the minimal unit in this process. As a result, there are some faces whose neighbouring patches have different resolutions. We do not allow refinement transition of multiple levels in the code, therefore the difference in patch length can only be a factor of three. As all patches in the domain have the same number of volumes, we shall have $\delta x_{\mathrm{coarse}} = 3\delta x_{\mathrm{fine}}$ as well.

The patches on the two sides of the boundary of the refinement transition both have face structures on their own resolution level, whose halves close to themselves can be copied from their own edges as usual. However, the out halves can not be copied straightforwardly this time, as their neighbours now hold volumes in a different resolution. That is the reason why we implement the so-called refinement transition strategy in the code. In the following content, we shall call the calculation scheme determining fine volumes from the coarse volumes as *Interpolation*, while calling the one to determine coarse volumes from fine volumes as *Restriction*.

We illustrate an example of the refinement transition setup between two face structures on the AMR boundary in figure 2.6, with a patch size of p and a halo of 3 layers. The two face structures, therefore, both have $6 \times p \times p$ volumes in total. The faces of the coarse patch and fine patch are plotted on the two sides of the figure, with one dimension omitted. The layers are labelled from c0 to c5 and f0 to f5 for clarity, and it does not represent the actual enumeration in the code implementation.

For the interpolation scheme, our job is to compute the outer half (light blue part) of the fine patch face, i.e., layers f0, f1 and f2, using the information of the volumes in the whole face of the coarse patch. Mathematically speaking, we are looking for a scheme mapping $6p^2$ variables to $3 \times (3p)^2$ variables. We can, in principle, do the mapping directly and create a matrix of size $27p^2 \times 6p^2$. However, most of the elements of this matrix would be zero, especially for low-order schemes due to their block-type nature. It thus leads to a waste of memory and computational resources. Therefore, we adopt the so-called tensor product approach to perform the mapping. The idea here is to decompose the high-dimensional map into the product of the

Figure 2.6: The illustration of the face structures on the coarse and fine side of a refinement transition boundary. The mapping between them determines the strategy of the refinement transition in the application. For the interpolation, we need to derive the outer half of the fine face (the light red squares on the right) using the information of the coarse face. On the other hand, we need to compute the outer half of the coarse face (the light blue squares on the left) utilising the fine face for a restriction scheme. The layers of the faces are labelled from c0 to c5 and f0 to f5 respectively, it is not the enumeration we employed in the actual code.

one-dimensional maps. Assuming the volumes in the coarse and fine faces are $Q^c_{i,j,k}$ and $Q^f_{i,j,k}$, the interpolation map can be written as

$$Q^f_{i,j,k} = P^x_{il} P^y_{jm} P^z_{kn} Q^c_{l,m,n}, \tag{2.43}$$

where P^x, P^y, and P^z are the matrices responsible for the corresponding one-dimensional map. As we shall have a symmetry for the two dimensions parallel to the face normal, the two matrices for mappings along them are identical. In the following content, we label the matrix for the mappings parallel to the face normal as $P^{\parallel}$ and the one for the map along the face normal as $P^{\perp}$.

In the current code, we implement two sets of matrices $P^{\parallel}$ and $P^{\perp}$ with different orders in accuracy. Matrix $P^{\parallel}$ should have a size of $3p \times p$, as it maps p coarse volumes to $3p$ fine volumes; on the other hand, matrix $P^{\perp}$ has a size of 3×6, as it maps 6 coarse layers to 3 fine layers (remember only the outer half of the fine face needs the interpolation). The first set of matrices $P^{\parallel}$ and $P^{\perp}$ is for the piecewise constant interpolation:

$$P^{\parallel}_{\text{const}} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & \ldots \\ 1 & 0 & 0 & 0 & 0 & \ldots \\ 1 & 0 & 0 & 0 & 0 & \ldots \\ 0 & 1 & 0 & 0 & 0 & \ldots \\ 0 & 1 & 0 & 0 & 0 & \ldots \\ 0 & 1 & 0 & 0 & 0 & \ldots \\ & & \ldots. & & & \end{pmatrix}, \; P^{\perp}_{\text{const}} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}. \tag{2.44}$$

That means we use the value in the coarse volume (layer) if the targeted fine volume (layer) is within it. It is the most straightforward interpolation one may have.

The second set of interpolation matrices is for the linear interpolation:

$$P^{\|}_{\text{linear}} = \begin{pmatrix} 4/3 & -1/3 & 0 & 0 & 0 & ... \\ 1 & 0 & 0 & 0 & 0 & ... \\ 2/3 & 1/3 & 0 & 0 & 0 & ... \\ 1/3 & 2/3 & 0 & 0 & 0 & ... \\ 0 & 1 & 0 & 0 & 0 & ... \\ 0 & 2/3 & 1/3 & 0 & 0 & ... \\ & & .... & & & \end{pmatrix}, \quad P^{\perp}_{\text{linear}} = \begin{pmatrix} 0 & 1/3 & 2/3 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2/3 & 1/3 & 0 & 0 \end{pmatrix}. \tag{2.45}$$

In this scheme, we apply a simple linear interpolation to compute the value in the fine volumes according to the relative positions of the closest two coarse volumes. The extrapolation is used for those volumes at the edge of the considered fine face. There is no such requirement in the $P^{\perp}$ as all three layers of the fine face are inside the coarse face.

The restriction, which is the reverse procedure of the interpolation, involves a mapping from the whole fine face to the outer half of the coarse face (light blue part), i.e., the layers c3, c4 and c5. Again, we adopt the tensor product for the restriction scheme, which decomposes the matrix of size $3p^2 \times 54p^2$ into three one-dimensional matrices. The tensor product formulation of restriction read as

$$Q^c_{i,j,k} = R^x_{il} R^y_{jm} R^z_{kn} Q^f_{l,m,n}, \tag{2.46}$$

where we label the restriction matrix as R. Similarly, we define the matrix $R^{\|}$ for the mappings parallel to the face normal and matrix $R^{\perp}$ for the mappings along the face normal.

In the code practices, we incorporate two versions of matrix $R^{\|}$: one for the injection (constant) scheme and the other for the average scheme. However, only one version of $R^{\perp}$ is implemented in the code as the corresponding scheme of zero-order accuracy yields significant numerical errors.

The matrix $R^{\|}$ for the injection scheme is

$$R^{\|}_{\text{injection}} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & ... \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & ... \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & ... \\ & & & & & ... & & & & & & \end{pmatrix} \tag{2.47}$$

In this scheme, we utilise the value in the central volume per three fine volumes for the coarse volume, as this central volume has the same central coordinate (in this parallel direction) as the restricted coarse volume. The matrix $R^{\|}$ for the average scheme is

$$R^{\|}_{\text{average}} = \begin{pmatrix} 1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... \\ 0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 0 & ... \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 & 0 & ... \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 1/3 & ... \\ & & & & & ... & & & & & & \end{pmatrix} \tag{2.48}$$

The average of the three corresponding fine volumes is used to assign the coarse volume and provide improved accuracy.

The version of matrix $R^{\perp}$ implemented in the code is

$$R^{\perp} = \begin{pmatrix} 0 & 0 & 0 & 1/3 & 1/3 & 1/3 \\ 0 & 0 & 0 & 0 & -2 & 3 \\ 0 & 0 & 0 & 0 & -5 & 6 \end{pmatrix}. \tag{2.49}$$

This matrix uses a combination of average and extrapolation for the map along the face normal. Notice the enumeration depends on the face orientation rather than relative position. So the right half of the matrix instead of the left half represents the inner part of the fine face (layers f3, f4 and f5).

The tensor product scheme we have described provides a quite convenient approach to implement refinement transition in low order accuracy. By enlarging the stencil, we can further increase the interpolation order, with a cost of computational overhead. In our current simulation test, low-order interpolation and average restriction have already demonstrated rather smooth refinement transitions at the AMR boundaries. However, some issues of inaccuracy and fluctuations still exist, especially when we apply a linear scheme to scenarios with spherical symmetry. A higher-order refinement transition scheme may help to solve those challenges. We refer readers to the corresponding section in the application chapters for discussions on them.

Besides an extension of the current tensor product scheme, another method for achieving a high-order refinement transition is to use the big three-dimensional matrix idea above, which maps the volumes between coarse and fine faces in three dimensions directly. It has the unique advantage of having access to the diagonal neighbours of considered volume. Compared to the cross-shape stencil of the tensor product scheme, the three-dimensional matrix employs a cubic stencil, thus reducing the size of the stencil significantly to reach the same order of accuracy as the tensor product. One possible scheme utilising the big matrix idea is the algorithm introduced by McCorquodale and Colella, 2011. which assumes a Taylor expansion around the considered volume. Rather than calculating the derivatives as coefficients directly, it takes the coordinates and values in the neighbour volumes and employs a constrained linear least-squares problem to fit the coefficients of expansions. The values in the targeted find/coarse volumes are then computed by substituting the coordinates into the Taylor expansion. The implementation of this scheme in ExaHyPE2 is currently in progress.

2.5 Particle Tracers as Data Probes

The last feature we introduced in this technical chapter is the tracer module. It is based on the particle support of ExaHyPE2 (Weinzierl et al., 2016). Originally, the particles have been introduced to support Particle-in-Cell codes. Nowadays, we use the particle feature as the baseline for the smoothed-particle hydrodynamics (SPH) code, where the particles themselves constitute the evolving subjects, and the

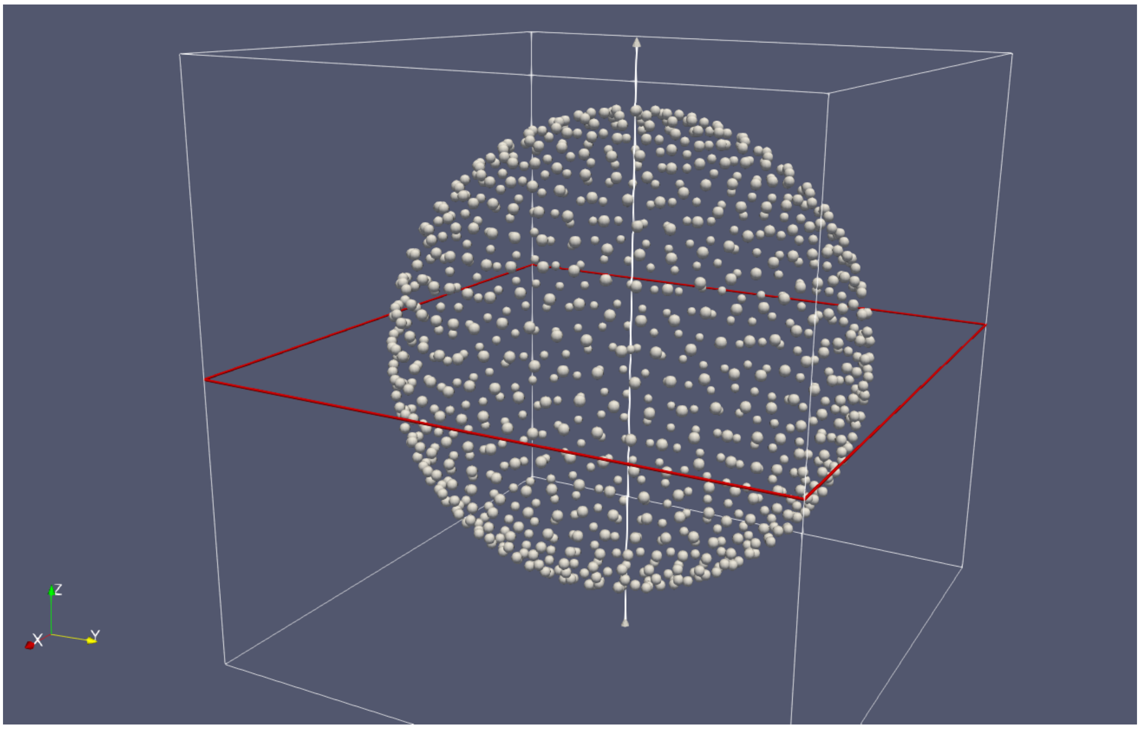

Figure 2.7: An example of a set of initialized tracers in ExaHyPE2. The tracers are illustrated as those white spheres in the domain. The set of tracers in this example is used to perform spherical integrals in our numerical relativity application, see section 4.1.5.

actual physical quantities are approximated based on the properties and interactions of individual particles.

In our present projects, however, we employ the particles for a different purpose, namely data probing or object tracking. Unlike in the SPH approach, the particles are not evolving subjects anymore, instead, they are associated with the grid structure we described above and record the evolving variables during the timestepping. Those particles allow users to examine the evolutions at certain points in more detail, and provide trajectory information on important objects in the domain. That is where the name "tracer" comes from, as they trace and track important features throughout the simulations.

To incorporate the tracers into the application, one needs to declare particles in the code and specify their behaviours. This procedure can be divided into three segments:

- *Initialization.* The first step to employing tracers is to initialize them, where the particles are added into and associated with the simulation grid. Users are asked to provide the number and initial coordinates of the tracers they would like to implement in the domain. Currently, ExaHyPE2 offers two approaches for initializing the coordinates of tracers: users can set the coordinates explicitly by writing them down in the Python script, or they can provide a coordinate file that the interface reads and assigns the coordinates accordingly. The latter approach is particularly efficient when injecting a large number of tracers into the domain. Users also need to specify what evolving variables the tracers need to plot. By default, the tracer would plot the whole evolving system which may lead to a waste of storage space. An example of initialized tracers is illustrated in figure 2.7. In this instance, we are implementing a group of tracers for sample points derived from the concept of *spherical t-design* (see section 4.1.5), these tracers are utilised to calculate the spherical integral in our numerical relativity application. A total of 948 tracers are deployed in the domain for this setup.

- *Data Projection.* As we mentioned, the particles do not enter the evolution and only record the values of quantities at their locations. One, therefore, needs to decide how the values in the volumes are projected to the tracers. Similar to what we do in Section 2.4.2, one choice is to project piece-wise constantly, which directly assigns the tracer with the value of the volume that the tracer is located. Another choice is to interpolate linearly utilising the values in the closest 8 (4 in two-dimensional setup) volumes, which is our default projection scheme in the current code. Another important feature users need to specify in this segment is the movement of the tracers. When projecting evolving quantities to the tracers, one can declare three certain quantities which are used by code as the velocity field v_i of the tracers themselves. If the velocity field is declared, the positions of the tracers are updated throughout the time evolving using an explicit Euler solver

$$x_i^{t+\delta t} = x_i^t + \delta t v_i^t, \tag{2.50}$$

where x_i^t is the position of the considered tracer at time t. That allows tracers to follow the vector field or flows in the simulation and the black hole tracker module in our numerical relativity application is based on this feature. We can, of course, leave the tracers at their initial places by not declaring the velocity field. In this case, those tracers are utilised as static data probes recording evolution at their positions.

- *Data dumping.* The last segment controls the data output of the tracers. By default, the tracers dump the information they record every timestep as arrays. The arrays appear as row entities in the final output file of the CSV format. For large-scale simulations, the tracer output can be massive and hard to process and visualized during analysis. Therefore, ExaHyPE2 offers various thresholds to control the dumping of the tracers: users can ask tracers to only dump at a fixed code time interval, or only dump when their recording variables or tracking positions show a substantial change. Users can also specify the output precision and set the maximal size limit for individual CSV files according to their needs. When the maximal size limit is reached, the code immediately writes the file into the disk and clears all the recorded arrays from previous timesteps in the memory. This approach helps to reduce the memory footprint of the tracer module significantly.

The tracer module is a rather flexible feature and is used in our applications extensively. However, we shall point out that the results we will report in the following application chapters do not heavily rely on this module. The primary focus of those chapters is the overall analysis and discussion of the simulation results, therefore will be mainly based on the global snapshots. Nevertheless, there are still some results that do make use of the tracer module, such as the black hole trackers, in our application. We refer readers to the corresponding section for more details on this.

CHAPTER 3

Application: Spherical Gas Accretion under the DGP Gravity

In this chapter, we present the application that we have implemented on ExaHyPE2 for simulating the spherical accretion scenario of gas. The physics background of the application is covered in section 3.1, including an introduction to the modified DGP model we use in the code and how self-similarities are achieved in both standard and modified gravity. The solution in the standard gravity is a review of the result of Bertschinger, 1985 and the theoretical derivation for the modified gravity is completed by Baojiu Li. We then describe the code implementation of our application, in section 3.2, focusing on the specialised treatment designed for the spherical collapse scenario and how they contribute to accomplishing a stable and well-behaved evolution. The test results are illustrated in section 3.3 and a discussion of the theoretical consideration and existing issues is given in section 3.4. The work in these three sections is completed by the author.

The content in the chapter is mainly based on the published paper: Han Zhang et al. "Spherical accretion of collisional gas in modified gravity I: self-similar solutions and a new cosmological hydrodynamical code". *Monthly Notices of the Royal Astronomical Society* 515.2 (2022), pp. 2464–2482.

3.1 Theoretical Background

We present the physics behind the spherical collapse scenario we investigated and implemented in the code in this section. We first introduce the general DGP gravity. To achieve self-similarity, we need to apply slight modifications to the original formulation. The test model we derive from it accordingly is discussed. We also give a brief review of the self-similar solution in an Einstein-de Sitter universe discovered by Bertschinger, 1985 and derive the self-similar solution for the modified version of the DGP model following a similar approach. We then compare the behaviour of the solutions in this model, for several different parameter choices, with that of the EdS model. This analysis reveals some interesting features of the solutions, which will be discussed in the corresponding section.

Throughout this paper, we assume that the background cosmology is that of the Einstein-de Sitter (EdS) universe, i.e., a flat matter-dominated background(for simplicity we assume that this still holds even in the DGP models). It used to be the standard cosmological model before the ΛCDM model and replaced it in the face of growing evidence that the cosmic expansion rate has been accelerating at late times, and it still serves as a good approximation for the real Universe between redshifts $\simeq 300$ and $\simeq 2$. The EdS universe assumes a zero cosmological constant and flat spatial curvature, and the equation of state of its non-relativistic matter content is $P(\rho) = 0$. With these parameters, the evolution of the scale factor of the universe, a, can be derived analytically from the Friedmann equation as $a(t) = Ct^{2/3}$, where t is the cosmic time, $C \equiv t_0^{-2/3}$ is a constant and t_0 is the cosmic time today (when $a = 1$). This is an important assumption we will use to derive the self-similar solution later. No specific unit system is used in this application, and we discuss the conversion between code units and physics units in section 3.2.1.

3.1.1 The DGP model

The Dvali-Gabadadze-Porrati (DGP) braneworld model is a modified gravity model in spacetime with an extra, fifth, dimension. The base assumption of this model is that the universe is a four-dimensional "brane" embedded in a five-dimensional spacetime, which is called a “bulk”.

This model provides an explanation as to why gravity is much weaker than other fundamental forces: all matter components are assumed to be confined on the brane, while gravitons could propagate through, or leak into, the extra spatial dimension.

The spacetime action of the DGP model is given by

$$S = \int_{\text{brane}} \mathrm{d}^4x\sqrt{-g}\frac{R}{16\pi G} + \int_{\text{bulk}} \mathrm{d}^5x\sqrt{-g^{(5)}}\frac{R^{(5)}}{16\pi G^{(5)}}, \tag{3.1}$$

where R is the Ricci scalar, g is the determinant of the metric tensor, G is Newton's constant, and the superscript $^{(5)}$ means the corresponding quantities live in the five-dimensional bulk. Others without it are normal four-dimensional quantities.

The modified Einstein equation for the DGP models can be derived from the variation of the gravitational action equation (3.1), which further leads to the following modified Friedmann equation that governs the cosmic expansion history $H(a)$:

$$\frac{H(a)}{H_0} = \sqrt{\Omega_{\mathrm{m}0}a^{-3} + \Omega_{\mathrm{DE}}(a) + \Omega_{\mathrm{rc}}} \pm \sqrt{\Omega_{\mathrm{rc}}}, \tag{3.2}$$

where $H_0 = H(a = 1)$ is the Hubble constant today (when the scale factor is $a = 1$), $\Omega_{\mathrm{m}0}$ is the present-day density parameter of matter (we have neglected the presence of radiation and massive neutrinos here since they are not relevant for the interest of this work), $\Omega_{\mathrm{DE}}(a)$ represents the density parameter of a possible additional dark energy species at time a, and $\Omega_{\mathrm{rc}} \equiv c^2/(4H_0^2r_c^2)$. Here, r_c is the so-called crossover scale, which is a new free model parameter that indicates the scale above which the gravity begins to deviate from the standard Einsteinian:

$$r_c \equiv \frac{1}{2}\frac{G^{(5)}}{G}. \tag{3.3}$$

It is easy to see that, equation (3.2) goes back to the usual form of the Friedmann equation when $H_0r_c \rightarrow \infty$.

The ± in equation (3.2) shows that this model has two branches of solutions. There is a “self-accelerating” branch (sDGP, the “+” branch) that can realise an accelerated Hubble expansion at late times without the need of a cosmological constant or dark energy, i.e., $\Omega_{\rm DE}(a) = 0$. However, this branch has several unsolved theoretical issues (Koyama, 2007). Additionally, its predicted cosmological history is significantly different from that of ΛCDM and the observation also disfavours this model (e.g., Song, Sawicki, and Hu, 2007).

The other branch, the so-called normal branch of DGP (nDGP) gravity, where equation (3.2) takes the "−" sign, can not provide an accelerated Hubble expansion by itself, and thus some additional dark energy component is needed ($\Omega_{\rm DE} \neq 0$) to explain the observation. This model has attracted much attention in recent years as it serves as a useful testbed of the Vainshtein screening mechanism (e.g., Brax, 2013), despite its unappealing property of being still in need for additional dark energy. We will describe Vainshtein screening in more detail below.

The (modified) Poisson equation of DGP gravity and corresponding equations of the scalar field has been derived by Koyama and Silva (2007):

$$\nabla^2\Phi = 4\pi G a^2 \delta\rho_{\rm m} + \frac{1}{2}\nabla^2\varphi\,, \tag{3.4}$$

and

$$\nabla^2\varphi + \frac{r_c^2}{3\beta\, a^2 c^2}\left[(\nabla^2\varphi)^2 - (\nabla_i\nabla_j\varphi)^2\right] = \frac{8\pi\, G\, a^2}{3\beta}\delta\rho_{\rm m}, \tag{3.5}$$

where Φ and φ are the gravitational potential and the scalar field of the model, respectively. They are also known as the brane-bending mode, which represents the position of the brane in the fifth dimension. ∇ is the spatial gradient (wrt to comoving coordinates), c is the speed of light and $\delta\rho_{\rm m} = \rho_{\rm m} - \bar{\rho}_{\rm m}$ is the matter density perturbation (throughout this paper an overbar denotes the background value of a quantity). β is a time-dependent function:

$$\beta(a) \equiv 1 \pm 2H\, r_c \left(1 + \frac{\dot{H}}{3H^2}\right), \tag{3.6}$$

for the two branches, which for the normal branch can be simplified as

$$\beta(a) = 1 + \frac{\Omega_{\rm m0}a^{-3} + 2\Omega_{\rm rc}}{2\sqrt{\Omega_{\rm rc}(\Omega_{\rm m0}a^{-3} + \Omega_{\rm rc})}}. \tag{3.7}$$

While we are interested in the DGP model, our main focus in this paper will be the effect of a *fifth force* that is mediated by the scalar field φ, denoted by the second term on the right-hand side of equation (3.4). To gain flexibility and to ensure self-similarity of the resulting model behaviour, we take the liberty to keep the main features of equation (3.5) but allow deviations from the exact behaviour of the sDGP or nDGP models. More explicitly, we will promote r_c to a time-dependent function, and also allow β to differ from equation (3.6). We remark that such variations from the original DGP model are not uncommon in other modified gravity models involving the Vainshtein mechanism, notably the cubic Galileon (Nicolis, Rattazzi, and Trincherini, 2009; Deffayet, Esposito-Farese, and Vikman, 2009) and the Proca (Heisenberg, 2014) theories. Our modification leads to losses of certain properties of the original DGP model, such as being an interesting alternative to Λ to explain the cosmic acceleration. Yet, the latter is not the focus on this work. In the context below, we call our modified DGP model the Self Similar Test (SST) model to indicate that this is a test case involving a modified gravity mechanism and self-similar property. This case is designed as a toy model achieving self-similarity in an enhanced and screening gravity and is used for code experiments of ExaHyPE2.

3.1.2 The Self-Similarity in Spherical Collapse

In this subsection, we describe the self-similar collapse of collisional and non-radiative gas in some models. We first review the classic result from Bertschinger (1985), which applies to standard gravity in EdS universe. Then we proceed to show that self-similarity can also be achieved in the SST model with Vainshtein screening. These can be used as a test case to verify our numerical implementation with ExaHyPE2 for both the standard and modified gravity scenarios, though our implementation of modified gravity is not restricted to the SST model where self-similarity holds.

Einstein-de Sitter universe

Consider a uniform spherical overdensity region in the matter-dominated universe background. Its initial condition can be written as

$$\rho = \frac{1}{6\pi G t_i^2}\begin{cases} 1+\delta_i, & r < R_i \\ 1, & r > R_i, \end{cases} \tag{3.8}$$

$t_i < t_0$ is the initial cosmic time for this scenario to begin, $\delta_i = \delta\rho/\bar{\rho} \ll 1$ the density contrast at t_i, where $\bar{\rho} = \bar{\rho}(t)$ and $\delta\rho$ are respectively the mean matter density at time t and the density perturbation, and R_i is the initial radius of the spherical overdensity region. At the beginning, the Hubble flow is approximately unperturbed as $\delta_i \ll 1$. Thus, we have $v_i = H_i r_i$ and $H_i = 2/(3t_i)$. As the universe expands, the matter inside R_i starts to decelerate and decouple from the Hubble flow because of the slightly higher density. At some point it stops expanding completely (so-called "turnaround") and turns into a collapse. The turnaround for the mass shell at R_i initially happens at a cosmic time and max radius (Bertschinger, 1985)

$$t_{\rm ita} = \frac{3\pi}{4}\delta_i^{-3/2} t_i, \quad r_{\rm ita} = R_i \delta_i^{-1}, \tag{3.9}$$

where the subscript $_{\rm ita}$ stands for "initial turnaround". Matter inside the initial overdensity region starts to collapse first and all matter there infalls at the same time. No shell crossing happens. The matter initially in more distant shells (i.e., at initial radii $r_i > R_i$) will start to collapse in progressively later times. The radius at which they turn around can be calculated using the Lagrangian picture. For the mass element initially located at r_i, its evolution obeys the Newton's gravity law:

$$\frac{\mathrm{d}^2 r}{\mathrm{d}t^2} = -\frac{Gm}{r^2}. \tag{3.10}$$

Here m accounts all mass interior to the shell we are considering. As no shell crossing happens during the evolution, it can be written as

$$m = m\left(r_i\right) = \frac{4}{3}\pi\rho_i r_i^3\left(1+\delta_i\frac{R_i^3}{r_i^3}\right) \equiv \frac{4}{3}\pi\rho_i r_i^3(1+\Delta), \tag{3.11}$$

where $\rho_i = 1/(6\pi G t_i^2)$ is the background density at t_i for an Einstein-de Sitter universe. It also defines $\Delta \equiv \delta_i R_i^3/r_i^3$. The equation (3.10) then can be recast using the following dimensionless time and radius variables, $\tau \equiv t/t_i$ and $y \equiv r/r_i$, as

$$\frac{\mathrm{d}^2 y}{\mathrm{d}\tau^2} = -\frac{2}{9}\left(1+\Delta\right)\frac{1}{y^2}. \tag{3.12}$$

Integrating this equation twice and using the assumption $\Delta \ll 1$, the solution can be expressed implicitly as (Bertschinger, 1985)

$$\tau = \frac{3}{4}\left(\theta - \sin\theta\right)\Delta^{-3/2} \equiv d\Delta^{-3/2}, \tag{3.13}$$

with

$$y\Delta = \sin^2\frac{\theta}{2} \equiv \eta, \tag{3.14}$$

where we have defined the variables d and η for later use. As turnaround happens when y reaches its maximum, this yields to $\theta_{\mathrm{ta}} = \pi$ (where a subscript $_{\mathrm{ta}}$ means "turnaround"). From equation (3.13) this corresponds to a time $\tau = (3\pi/4)\Delta^{-3/2}$ and $y_{\mathrm{ta}} = r_{\mathrm{ta}}/r_i = \Delta^{-1}$. Combining these two expressions with the relationship between δ_i, Δ and R_i given in equation (3.11), it is straightforward to derive the following expression of the turnaround radius:

$$r_{\mathrm{ta}}(t) = \left(\frac{3\pi}{4}t_i\right)^{-8/9}\delta_i^{1/3}R_i t^{8/9}, \quad \text{for } t \geq t_{\mathrm{ita}}. \tag{3.15}$$

Now we switch to the fluid picture. The motion of a collisional gas in this system is governed by the gravity-driven Euler equations:

$$\begin{aligned}
\frac{\mathrm{d}\rho}{\mathrm{d}t} &\equiv \left[\frac{\partial}{\partial t} + v\frac{\partial}{\partial r}\right]\rho = -\rho\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2 v\right), && (3.16)\\
\frac{\mathrm{d}v}{\mathrm{d}t} &= -\frac{1}{\rho}\frac{\partial p}{\partial r} - \frac{Gm}{r^2}, && (3.17)\\
\frac{\mathrm{d}}{\mathrm{d}t}\left(p\rho^{-\gamma}\right) &= 0, && (3.18)\\
\frac{\partial m}{\partial r} &= 4\pi r^2\rho, && (3.19)
\end{aligned}$$

where $\rho = \rho(r,t)$, $v = v(r,t)$ and $p = p(r,t)$ are the density, velocity and pressure of the fluid at radius r and time t. $m \equiv m(< r)$ represents the total mass within a given radius r, and γ the adiabatic index. We now use equation (3.15) and define the new radial coordinate:

$$\lambda \equiv \frac{r}{r_{\rm ta}}, \tag{3.20}$$

as well as the dimensionless quantities V, D, P and M:

$$\begin{aligned} v(r,t) &= \frac{r_{\rm ta}}{t} V(\lambda), && (3.21)\\ \rho(r,t) &= \rho_{\rm H} D(\lambda), && (3.22)\\ p(r,t) &= \rho_{\rm H} \left(\frac{r_{\rm ta}}{t}\right)^2 P(\lambda), && (3.23)\\ m(r,t) &= \frac{4\pi}{3}\rho_{\rm H} r_{\rm ta}^3 M(\lambda), && (3.24) \end{aligned}$$

where $\rho_{\rm H} = \rho_{\rm H}(t)$ is the critical density at time t, which is equal to the mean matter density $\bar{\rho}_{\rm m}(t)$ in the EdS model. These allow us to cast equations (3.16) - (3.19) as the following new dimensionless fluid equations (Bertschinger, 1985):

$$\begin{aligned} \left(V - \frac{8}{9}\lambda\right) D' + DV' + 2\frac{D}{\lambda}V - 2D &= 0, && (3.25)\\ \left(V - \frac{8}{9}\lambda\right) V' - \frac{1}{9}V &= -\frac{P'}{D} - \frac{2}{9}\frac{M}{\lambda^2}, && (3.26)\\ \left(V - \frac{8}{9}\lambda\right)\left(\frac{P'}{P} - \gamma\frac{D'}{D}\right) &= \frac{20}{9} - 2\gamma, && (3.27)\\ M' &= 3\lambda^2 D, && (3.28) \end{aligned}$$

where a prime means the derivative wrt λ. Those equations only have one variable λ and thus could be solved directly given proper boundary conditions (see Section 3.1.3 below). No further time or length scales are involved, which means that the solutions to the system would remain identical throughout the evolution if expressed in terms of the λ coordinate. This is where self-similarity comes from. Obviously, if any new terms added in equations (3.16) - (3.19) depend on other scales besides λ, the solution to that new system will deviate from this self-similar solution.

SST model

In the spherically symmetric system, the scalar field equation (3.5) gets simplified significantly (see, e.g., Li, Zhao, and Koyama, 2013):

$$\frac{2r_c^2}{3\beta c^2}\frac{1}{r^2}\frac{\partial}{\partial r}\left[r\left(\frac{\partial\varphi}{\partial r}\right)^2\right]+\frac{1}{r^2}\left[r^2\frac{\partial\varphi}{\partial r}\right]=\frac{8\pi G}{3\beta}\delta\rho_m. \tag{3.29}$$

This equation does not contain scale factor a as we use physics radius here. We then define

$$\hat{m}(r)\equiv 4\pi\int_0^r \delta\rho_m(r')r'^2\mathrm{d}r', \tag{3.30}$$

where we have used $\hat{m}$ to distinguish from $m(r)$ introduced in equation (3.10), since $\hat{m}$ does not account for the background matter density. equation (3.29) then can be integrated once to give:

$$\frac{2r_c^2}{3\beta c^2}\frac{1}{r}\left(\frac{\partial\varphi}{\partial r}\right)^2+\frac{\partial\varphi}{\partial r}=\frac{2}{3\beta}\frac{G\hat{m}(r)}{r^2}\equiv\frac{2}{3\beta}g_{\mathrm{N}}(r), \tag{3.31}$$

solving which gives the radial gradient of the scalar field directly as

$$\frac{\partial\varphi}{\partial r}=\left[2\frac{2r_c^2}{3\beta c^2}\frac{1}{r}\right]^{-1}\left[-1+\sqrt{1+4\frac{2r_c^2}{3\beta c^2}\frac{1}{r}\frac{2}{3\beta}g_{\mathrm{N}}}\right], \tag{3.32}$$

where we have dropped the other branch of solution that is unphysical. The equation could be simplified further by defining the "Vainshtein radius" r_V as follows:

$$r_V^3=\frac{16r_c^2}{9\beta^2c^2}G\hat{m}\left(r_{\mathrm{ta}}\right). \tag{3.33}$$

Note that here we have used $\hat{m}$ within $r_{\mathrm{ta}}(t)$ to define the Vainshtein radius, which differs from the usual definition that only accounts for the mass within the tophat radius — this is for convenience because in this way we end up with a generic expression that does not depend on the particular size of any tophat. Now the gradient of the scalar field reads as:

$$\frac{\partial\varphi}{\partial r}=\frac{4}{3\beta}\frac{r^3}{r_V^3}\frac{\hat{m}\left(r_{\mathrm{ta}}\right)}{\hat{m}(r,t)}\left[\sqrt{1+\frac{r_V^3}{r^3}\frac{\hat{m}(r,t)}{\hat{m}\left(r_{\mathrm{ta}}\right)}}-1\right]g_{\mathrm{N}}(r). \tag{3.34}$$

Note that $\partial\varphi/\partial r$ determines the strength of the fifth force, and one can easily see the following limiting behaviour:

$$\begin{aligned} \frac{\partial\varphi}{\partial r} &\simeq \frac{2}{3\beta}g_{\mathrm{N}}(r), \qquad r \gg r_V, \\ \frac{\partial\varphi}{\partial r} &\ll \frac{2}{3\beta}g_{\mathrm{N}}(r), \qquad r \ll r_V. \end{aligned} \tag{3.35}$$

If the scale of studied problem is significantly smaller than the Vainshtein radius r_V, the gradient of the scalar field is also much smaller than that of the Newtonian potential, such that the fifth force is negligible compared with the standard Newtonian force. This is the idea behind the Vainshtein screening.

Our next step is to try to recast the expression of the fifth force in the self-similar form (which, needless to say, is not always possible) similar to what we get above for the Einstein-de Sitter universe. This means that we hope that the ratio between the fifth force and standard Newtonian gravity, i.e., the coefficient in front of equation (3.34), depends on time t and radius r only through the combination $r_{\mathrm{ta}}(t)$. We again define $\lambda \equiv r/r_{\mathrm{ta}}$; note that this r_{ta} is the same as in equation (3.15))— this is mainly for convenience, but it does mean the r_{ta} in this expression is no longer the true turnaround radius in the SST model. The mass can then be rewritten, using the definition of $M(\lambda)$ given in equation (3.24), as

$$\hat{m}(r,t) = \frac{4\pi}{3}\rho_{\mathrm{H}} r_{\mathrm{ta}}^3 \left[M(\lambda) - \lambda^3\right] \equiv \frac{4\pi}{3}\rho_{\mathrm{H}} r_{\mathrm{ta}}^3 \hat{M}(\lambda). \tag{3.36}$$

As mentioned above, we have removed the contribution from the background mass as the fifth force only depends on density perturbations. The Vainshtein radius now reads as

$$r_V^3 = \frac{16 r_c^2}{9\beta^2 c^2}\frac{2}{9}\frac{r_{\mathrm{ta}}^3}{t^2}\hat{M}(1), \tag{3.37}$$

so that

$$\frac{r_V^3}{r^3} = \frac{16 r_c^2}{9\beta^2 c^2}\frac{2}{9t^2}\frac{\hat{M}(1)}{\lambda^3}, \tag{3.38}$$

and

$$\frac{\hat{m}(r,t)}{\hat{m}\left(r_{\mathrm{ta}}\right)} = \frac{\hat{M}(\lambda)}{\hat{M}(1)}. \tag{3.39}$$

These mean that

$$\frac{r_V^3}{r^3}\frac{\hat{m}(r,t)}{\hat{m}\left(r_{\mathrm{ta}}\right)}=\frac{16r_c^2}{9\beta^2c^2}\frac{2}{9t^2}\frac{\hat{M}(\lambda)}{\lambda^3}. \tag{3.40}$$

To achieve self-similarity, we need to ensure that equation (3.40) only depends on λ. The t dependence of r_c^2/β^2t^2 needs to be cancelled out. However, β also appears in equation (3.34) in the overall factor $4/(3\beta)$, and thus should be constant over time to avoid reintroducing an explicit t dependency. This then leads to $r_c \propto t \propto a^{3/2}$, with the second proportionality true in an Einstein-de Sitter universe.

Denoting $r_c(t)=r_{c0}(t/t_0)$, where t_0 is the cosmic time today and r_{c0} is the value of r_c at t_0, and defining the dimensionless constant

$$\zeta\equiv\frac{r_{c0}}{ct_0}=\frac{r_{c0}\times(t/t_0)}{ct}=\frac{r_c(t)}{ct}=\frac{3H(t)r_c(t)}{2c}=\frac{3H_0r_{c0}}{2c}, \tag{3.41}$$

we get

$$\frac{r_V^3}{r^3}\frac{\hat{m}(r,t)}{\hat{m}\left(r_{\mathrm{ta}}\right)}=\frac{32\zeta^2}{81\beta^2}\frac{\hat{M}(\lambda)}{\lambda^3}. \tag{3.42}$$

Therefore, the solution can be written as

$$\frac{\partial\varphi}{\partial r}=\frac{27\beta}{8\zeta^2}\frac{\lambda^3}{\hat{M}(\lambda)}\left[\sqrt{1+\frac{32\zeta^2}{81\beta^2}\frac{\hat{M}(\lambda)}{\lambda^3}}-1\right]g_{\mathrm{N}}(r). \tag{3.43}$$

This expression shows that the fifth-force-to-Newtonian-gravity ratio can be written in a form that only depends on λ, which satisfies the requirement of self-similarity. It is straightforward to show that the coefficient of g_{N} in the above equation is always smaller than $2/(3\beta)$, which means that the Vainshtein screening always works (though not necessarily always strong).

Let us briefly comment that, according to its definition in equation (3.41), ζ is the ratio between the crossover radius $r_c(t)$ and ct. The latter can be considered as some characterisation of the size of the Einstein-de Sitter universe (actually it is $3ct$). Therefore, the fact that this ratio is a constant in time implies that the Vainsthein screening mechanism is always effective on scales that correspond to a fixed fraction of the size of the universe, and therefore it should not be surprising that the self-similar properties of the EdS model have been preserved for this particular choice of

$r_c(t)$. Since r_c characterises the length scale beyond which gravity is modified in the SST model, we expect that for any physically interesting scenario we need to have $\zeta \sim \mathcal{O}(1)$. The choice of $\zeta = 2/3$, for example, corresponds to $H_0 r_{c0}/c = 1$, which leads to a similar Vainshtein screening efficiency to that for a typical parameter choice in studies of the nDGP model for the same value of β.

The actual strength of the fifth force is $\frac{1}{2}\frac{\partial\varphi}{\partial r}$, which means that the final expression for the fifth-force-to-Newtonian-gravity ratio is given by

$$\xi(\lambda) \equiv \frac{27\beta}{16\zeta^2}\frac{\lambda^3}{\hat{M}(\lambda)}\left[\sqrt{1+\frac{32\zeta^2}{81\beta^2}\frac{\hat{M}(\lambda)}{\lambda^3}}-1\right]. \tag{3.44}$$

Turning to the derivation of the self-similar equations in the SST model, i.e., the counterparts of equations (3.25) - (3.28), it is evident that only equation (3.26) needs to be modified. It is the only place where the law of gravity enters the calculation. However, instead of simply multiplying the $-\frac{2}{9}\frac{M}{\lambda^2}$ by $1+\xi(\lambda)$, the correct final version of equation (3.26) is slightly more complicated. This is because $\xi(\lambda)$ is the ratio between the fifth force and $g_{\rm N}$, which itself does not receive any contribution from the background matter density, c.f., equation (3.31). On the other hand, the term $-\frac{2}{9}\frac{M}{\lambda^2}$ contains contributions from the background matter. Taking this into account leads to the following SST version of equation (3.26):

$$\left(V-\frac{8}{9}\lambda\right)V'-\frac{1}{9}V = -\frac{P'}{D}-\frac{2}{9}\frac{M}{\lambda^2}-\frac{2}{9}\frac{\hat{M}}{\lambda^2}\xi(\lambda), \tag{3.45}$$

or equivalently

$$\left(V-\frac{8}{9}\lambda\right)V'-\frac{1}{9}V = -\frac{P'}{D}-\frac{2}{9}\frac{M}{\lambda^2}\left[1+\xi(\lambda)\right]+\frac{2}{9}\lambda\xi(\lambda). \tag{3.46}$$

A similar modification also appears in the SST counterpart of equation (3.10), which now reads

$$\ddot{y} = -\frac{2}{9}\left(1+\Delta\right)\frac{1}{y^2}\left[1+\xi\left(y,\tau\right)\right]+\frac{2}{9}\frac{y}{\tau^2}\xi(y,\tau), \tag{3.47}$$

where ξ has been defined in equation (3.44), but is now expressed in terms of the dimensionless radius and time, y and τ. More explicitly:

$$\xi = \frac{27\beta}{16\zeta^2}\frac{y^3}{(1+\Delta)\tau^2 - y^3}\left[\sqrt{1+\frac{32\zeta^2}{81\beta^2}\left(\frac{1+\Delta}{y^3}\tau^2 - 1\right)} - 1\right]. \tag{3.48}$$

This equation is needed for the exact solution of our equations in the next section.

Before concluding this subsection, let us note that one limit of the SST model arises from $\zeta \to 0$, in which equation (3.43) approaches

$$\frac{\partial\varphi}{\partial r} \to \frac{2}{3\beta}g_{\mathrm{N}}(r), \tag{3.49}$$

and so the fifth-force-to-Newtonian-gravity ratio approximately becomes $1/(3\beta)$, which is the linear-regime (i.e., no screening) solution. This corresponds to a time- and scale-independent enhancement of Newton's constant by a factor of $1/(3\beta)$ since we are assuming β to be a constant here.

3.1.3 Theoretical Solution of Self-Similar behaviour

Our next step is to find the exact solution to our self-similar equations equations (3.16) - (3.19): the profile of $D(\lambda)$, $V(\lambda)$, $P(\lambda)$ and $M(\lambda)$.

At the beginning stage, the spherical collapse can be described by a pressureless infall. Outside the radius of the tophat, the inner spherical shells infall at a greater speed than the outer shells, meaning that there is no shell-crossing or squeezing. However, when the infall speed of a given shell increases to a point where it exceeds the sound speed c_s of the fluid, the shell impacts upon the fluid element inside it before there is enough time for the latter to adjust. A discontinuity of fluid properties, such as velocity, pressure and density, then starts to arise there, which is known as a shock. The shock location is our primary quantity of interest when we validate the outcome of our simulation. We assume the shock happens at radius r_s or $\lambda_s \equiv r_s/r_{\mathrm{ta}}$ (the subscript $_s$ means shock), where we can apply the Rankine–Hugoniot jumping conditions, written in dimensionless forms:

$$D_2V_2 = D_1V_1 + (D_2 - D_1)\,V_s, \tag{3.50}$$

$$D_2V_2^2 + P_2 = D_1V_1^2 + P_1 + (D_2V_2 - D_1V_1)\,V_s, \tag{3.51}$$

$$\begin{aligned} D_2V_2\left(\frac{\gamma}{\gamma-1}\frac{P_2}{D_2} + \frac{1}{2}V_2^2\right) - D_1V_1\left(\frac{\gamma}{\gamma-1}\frac{P_1}{D_1} + \frac{1}{2}V_1^2\right) \\ = V_s\left[D_2\left(\frac{1}{\gamma-1}\frac{P_2}{D_2} + \frac{1}{2}V_2^2\right) - D_1\left(\frac{1}{\gamma-1}\frac{P_1}{D_1} + \frac{1}{2}V_1^2\right)\right]. \end{aligned} \tag{3.52}$$

Here, a subscript 1 or 2 is used to denote the preshock and postshock values of a quantity, respectively, and V_s is the dimensionless speed of the shock position itself. Physically, the three jumping conditions represent the continuity of mass, momentum and energy across the shock.

One can analytically calculate the preshock solutions in terms of λ_s using equation (3.12) and its solutions, equations (3.13) - (3.14) for $\Delta \ll 1$:

$$\begin{aligned} D_1 &= \frac{d_s^2\eta_s^{-3}}{1+3\chi_s}, && (3.53)\\ P_1 &= 0, && (3.54)\\ V_1 &= \lambda_s\frac{\sin\theta_s\,(\theta_s - \sin\theta_s)}{(1-\cos\theta_s)^2}, && (3.55)\\ M_1 &= \lambda_s^3 d_s^2\eta_s^{-3}. && (3.56) \end{aligned}$$

where $\theta_s = \theta\,(\tau_s)$, $\eta_s \equiv \sin^2\frac{\theta_s}{2} \equiv y_s\Delta$, $d_s \equiv \frac{3}{4}(\theta_s - \sin\theta_s)$ are the values of η and d at θ_s, and, $\chi_s \equiv 1 - \frac{3}{2}\frac{V_s}{\lambda_s}$. Combining equations (3.50) - (3.56), we get the boundary condition for the other side (post side) of the shock:

$$\begin{aligned} D_2 &= \frac{\gamma+1}{\gamma-1}D_1, && (3.57)\\ V_2 &= \frac{8}{9}\lambda_s + \frac{\gamma-1}{\gamma+1}\left(V_1 - \frac{8}{9}\lambda_s\right), && (3.58)\\ P_2 &= \frac{2}{\gamma+1}D_1\left(V_1 - \frac{8}{9}\lambda_s\right)^2, && (3.59)\\ M_2 &= M_1. && (3.60) \end{aligned}$$

The entire postshock solution can then be obtained by numerically integrating equations (3.25) - (3.28) inwards from $\lambda = \lambda_s$, using these boundary conditions. However, since λ_s is not known *a priori*, this is a trial and error process where the value of λ_s is updated iteratively until when the corresponding solutions meet the following

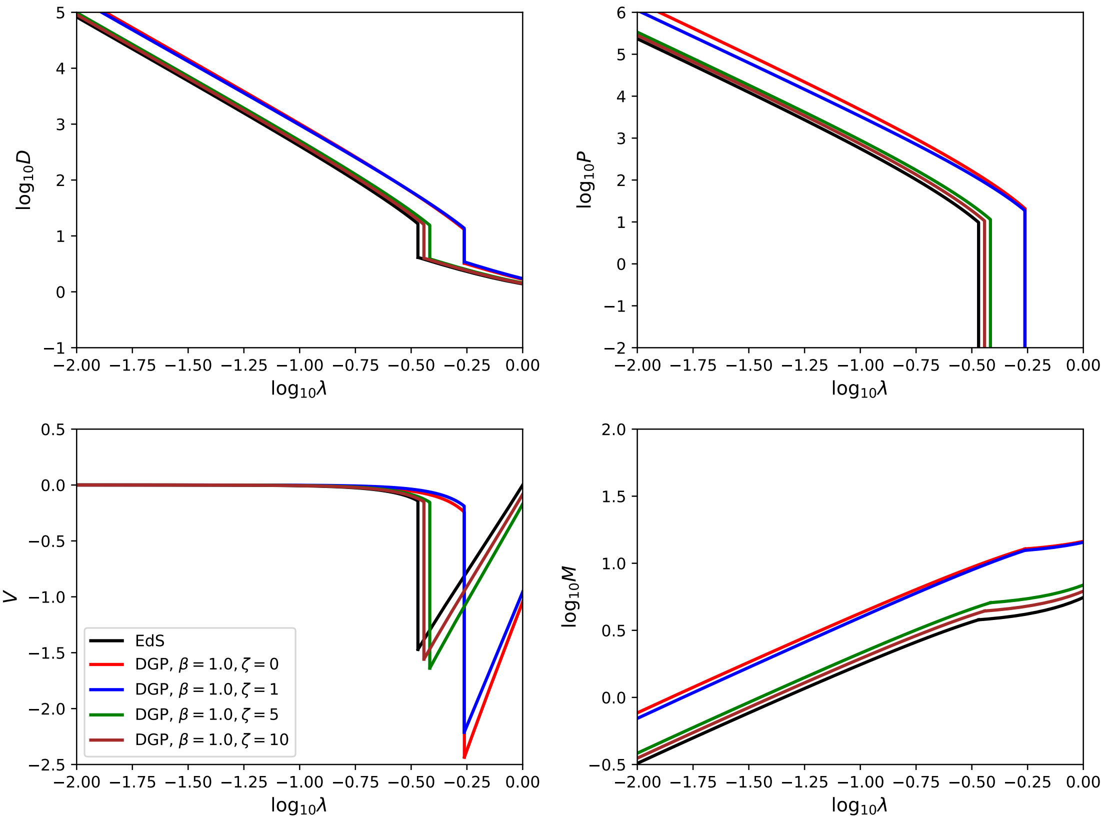

Figure 3.1: Self-similar solution for gravity in Einstein-de Sitter universe and SST models with different ζ choices. Rescaled $D(\lambda)$, $P(\lambda)$, $V(\lambda)$, $M(\lambda)$ are plotted. The value of ζ indicates the strength of the Vainshtein screening. It gets more efficient when ζ gets bigger. The case $\zeta = 0$ means there is no screening, i.e., The modification of SST are equivalent to a constant enhancement of gravity all the time. All curves here are obtained by using $\Delta = 0.001$.

physical boundary conditions in the centre of the system:

$$V(\lambda = 0) = M(\lambda = 0) = 0. \tag{3.61}$$

This is how Bertschinger, 1985 got his self-similar solution and we plot our reproduced result here in Figure 3.1.

While the use of the θ variable to write the solution to equation (3.12) in the implicit forms of equations (3.13) - (3.14) is convenient, this is impossible for the SST model where the corresponding spherical collapse equation takes a more complicated form. However, the introduction of θ in the EdS model is largely a matter of choice for convenience, and the same physics can be produced using τ as well. Because this is what we shall use for the SST model, we decide to also use τ instead of θ to obtain the numerical self-similar solutions for the EdS model. This means that we need to express the preshock solutions to D, P, V and M at τ_s. For the velocity, using its definition

$$v = \frac{\mathrm{d}r}{\mathrm{d}t} = \frac{r_i}{t_i}\dot{y} = \frac{r}{t}\frac{\tau}{y(\tau)}\dot{y} = \lambda\frac{r_{\mathrm{ta}}}{t}\frac{\tau}{y(\tau)}\dot{y}(\tau),$$

we obtain

$$V_1(\lambda_s) = \lambda_s\frac{\tau_s}{y\left(\tau_s\right)}\dot{y}_s, \tag{3.62}$$

where an overdot denotes the derivative wrt τ, and $\dot{y}_s = \dot{y}\left(\tau_s\right)$. For M, using

$$M = \frac{m}{\frac{4}{3}\rho_{\mathrm{H}} r_{\mathrm{ta}}^3} = \frac{\rho_i}{\rho_{\mathrm{H}}}\left(\frac{r_i}{r_{\mathrm{ta}}}\right)^3(1+\Delta),$$

we have

$$M_1(\lambda_s) = \left(\frac{3\pi}{4}\right)^{8/3}\frac{1+\Delta}{\Delta}\tau_s^{-2/3}. \tag{3.63}$$

For D, using

$$3\lambda^2 D(\lambda) = \frac{\mathrm{d}M/\mathrm{d}\tau}{\mathrm{d}\lambda/\mathrm{d}\tau} = \frac{-\frac{2}{3}\left(\frac{3\pi}{4}\right)^{8/3}\frac{1+\Delta}{\Delta}\tau^{-5/3}}{\left(\frac{3\pi}{4}\right)^{8/9}\Delta^{-1/3}\tau^{-8/9}\left(\dot{y} - \frac{8}{9}\frac{y(\tau)}{\tau}\right)},$$

we have

$$D_1(\lambda_s) = -\frac{2}{9}\tau_s\frac{1+\Delta}{y_s^2\left(\dot{y}_s - \frac{8}{9}\frac{y_s}{\tau_s}\right)}. \tag{3.64}$$

For P, we have $P_1 = 0$ again.

The following steps are the same as before. We can use the Rankine–Hugoniot jumping conditions to obtain D_2, P_2, V_2 and M_2, and numerically integrate the equations again to find the postshock solutions. This time we need to vary τ_s for our trial-and-error process after Δ is specified. y_s and $\dot{y}_s$ can be calculated numerically from τ by using equation (3.10) for the EdS model and equation (3.47) for the SST model. For EdS, we have explicitly checked that using the τ-based approach to set up the boundary conditions for the postshock solutions gives the identical answer as using equations (3.53) - (3.56), as expected.

We summarise our result for self-similar solutions in SST gravity within Figure 3.1. The black curves in the figure are the self-similar solutions to D, P, V and M for the EdS model, which we find to be in excellent agreement with literature results (e.g., Bertschinger, 1985). The coloured curves show the results for several variants of the SST model described in Section 3.1.2, with the case $\zeta = 0$ (red) corresponding to a constant enhancement of G by $1/(3\beta)$. The cases with $\zeta = 1$, 5 and 10 represent progressively more efficient Vainshtein screening, which explains why they are in between the EdS and $\zeta = 0$ cases. In particular, we see that at $\zeta = 10$ the screening is already very efficient so that the brown curves are very close to EdS. The qualitative trend also agrees with what one should expect for a model with enhanced gravity: the infall becomes faster such that the preshock solution of V becomes more negative and the shock happens at a larger radius; the density D and pressure P are also higher due to the stronger structure formation, and the latter explains why the enclosed mass M within a given radius is larger.

The SST results in Figure 3.1 are obtained with the parameter $\Delta = 0.001$. The Δ dependence of the solution, which we have explicitly checked, can already be seen at the equation level, cf. equations (3.62) -(3.64), and also in Fig. 3.2, where we show the two sets of self-similar solutions for two models, EdS (dashed lines) and SST with $\beta = 1.0, \zeta = 1$ (solid lines). Different colours indicate the value of Δ for each curve, as shown by the legends. We can see that the SST model has a very strong Δ dependence. A similar dependence is also present in the EdS case, but is much weaker there — indeed, it is known that in EdS there is approximately no Δ dependence in the limit $\Delta \ll 1$ (Bertschinger, 1985). This Δ dependence comes from our choice of using the turnaround radius $r_{\mathrm{ta}}(\tau)$, given in equation (3.15), to define the dimensionless coordinate λ, where r_{ta} itself depends on Δ. For the EdS model, rescaling r using this turnaround radius helps to cancel out the Δ dependence from equation (3.12), because this r_{ta} is calculated from the same dynamical equation and has the physical meaning of where the shell starts to collapse. But such a cancellation should not be expected to happen when we use the same equation (3.15) to define λ for the SST (and generally other gravity) models, since it does not represent the true turnaround radius anymore.

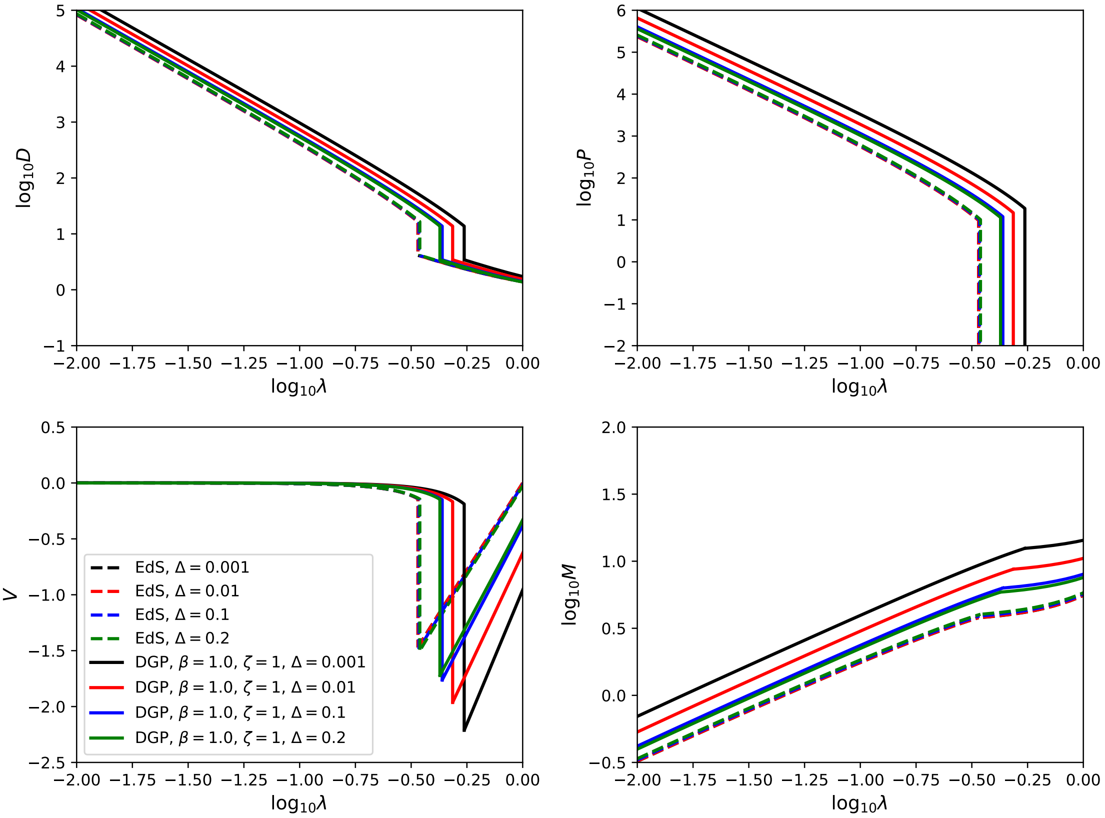

Figure 3.2: Two sets of the self-similar solutions are presented here, with the dashed lines for standard gravity in EdS and solid lines for SST model of $\beta = 1.0$, $\zeta = 1$. Different choices of Δ are indicated by colours. The Δ dependence in EdS case is negligible as the solutions assume the limit $\Delta \ll 1$ (Bertschinger, 1985). On the other hand, the results for SST model shows a clear dependence on Δ, which we explain in the text.

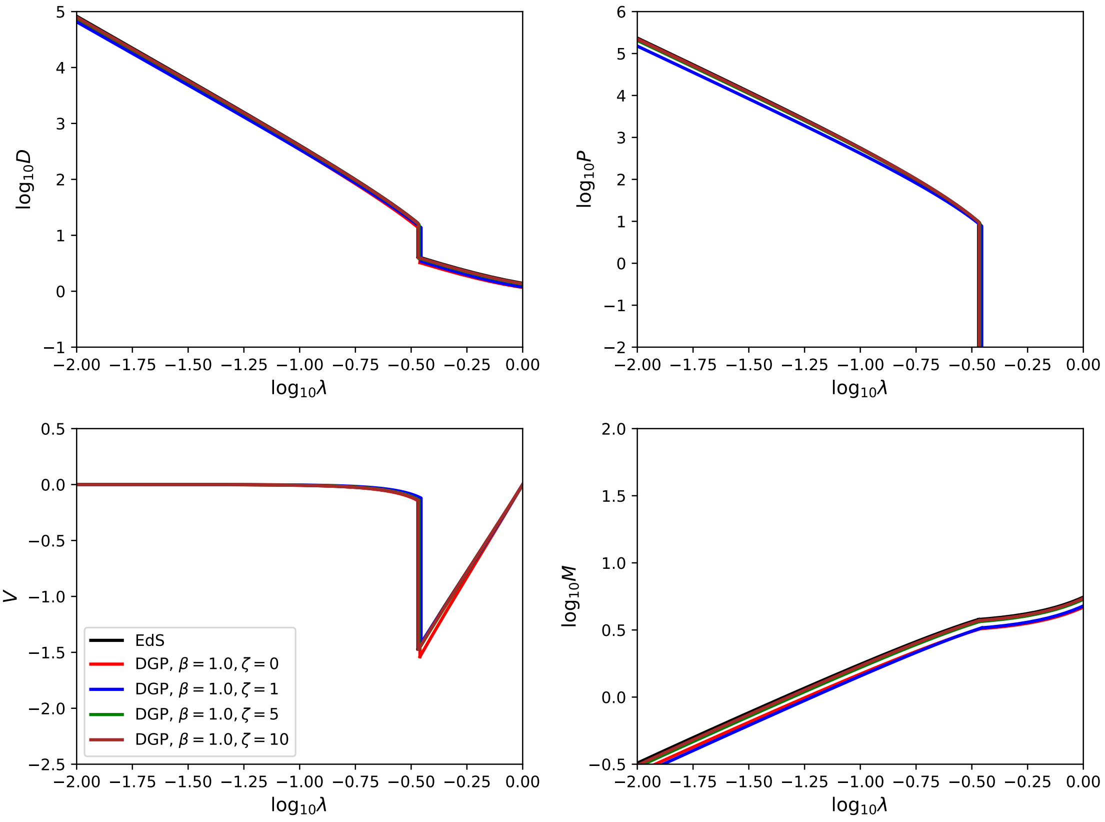

Figure 3.3: The new self-similar solutions of SST models with the same ζ choices as in previous figures, after the 're-rescaling'. Those curves now have little dependence on Δ, and behave closely to the EdS cases.

Using the same r_{ta} to define λ in all models above certainly has its advantages. One of these is that equation (3.15) is an analytical function with a power-law dependence on τ, which is convenient when deriving the dimensionless equations governing the self-similar evolution. It also allows these equations to take a similar form between the SST and EdS models. For example, equations (3.25) - 3.28) remain almost the same for the SST model, with only some slight changes of equation (3.26) to equation (3.46). In addition, Figure 3.1 clearly shows the effect of modified gravity law on the collapse of collisional gas and on the formation of shock: this also benefits from the fact that we have used the *same* 'turnaround' radius, $r_{\text{ta}}(\tau)$, to define the rescaled quantities in *all* models, so that the differences in the rescaled quantities reflect the differences in the same quantities pre-rescaling. Nevertheless, for theoretical interest, we also want to see the results when we actually define λ using the *true* turnaround radius of each model. Because there are no analytical expressions for r_{ta} for the SST model, this has to be done in a "post-processing" way: after getting the profile $V(\lambda)$ by following the above steps, we can obtain the real turnaround radius in the preshock $V(\lambda)$ solution, by looking for the value of λ'_{ta} where $V(\lambda'_{\text{ta}})$ crosses 0; we then get the correct turnaround radius as:

$$r'_{\text{ta}} = \left(\frac{\lambda'_{\text{ta}}}{\lambda_{\text{ta}}}\right) r_{\text{ta}} \equiv \alpha r_{\text{ta}}, \tag{3.65}$$

and use r'_{ta} to rescale our solutions for the other quantities, which is equivalent to performing the following 're-rescaling':

$$\begin{aligned}
\lambda &\rightarrow \alpha^{-1}\lambda, \\
D &\rightarrow D, \\
P &\rightarrow \alpha^{-2}P, \\
V &\rightarrow \alpha^{-1}V, \\
M &\rightarrow \alpha^{-3}M.
\end{aligned} \tag{3.66}$$

The new result is summarised in Figure 3.3. While we only show the results obtained by using $\Delta = 0.001$ here, we find that using $\Delta = 0.01,\ 0.1,\ 0.2$ give very similar results. One notable property is that the new rescaled profiles are very close to that in Einstein-de Sitter universe, i.e., the SST model behaves similarly to standard gravity if expressed in terms of the λ coordinate which is defined using the true turnaround radius of the model. As the real physical evolutions of these models are very different, this similarity is quite interesting, since it suggests that self-similarity works (at least to a good approximation) in more general models than just EdS.

As we shall see below, this "re-rescaling" idea using the true turnaround radius can also be applied to the numerical simulation result from ExaHyPE 2, and help to check its reliability in handling this scenario.

3.2 Code Implementation

We present how we configure the spherical collapse scenario on the ExaHyPE2 in this section. We will not cover the general technical details of building applications here as they have been covered in chapter 2. This section focuses on the code designs and implementation specialised for this application.

Compared to our Numerical Relativity application that will be introduced in chapter 4, the spherical collapse application is rather simple, but still require special treatments to achieve a stable and well-behaved simulation. Several techniques we reported in chapter 2 are designed and implemented driven by the development of this application. We are currently using the Rusanov Finite Volume solver (Section 2.2.1) with AMR support of ExaHyPE2. The resolution transition (interpolation and restriction) utilizes the tensor product scheme discussed in Section 2.4.2 with an accuracy of linear. The volume mass counting and the spherical mass interpolation are implemented using the Code Extension feature introduced in Section 2.1. This application also supports the particle tracer module (section 2.5) of ExaHyPE2, but our results reported in section 3.3 do not rely on this.

In the content below, we first discuss the code unit system adopted in our spherical collapse application. We then outline the simulation setups that we used to produce the reported results. Following that, we dedicate a separate section focusing on the implementation of initial and boundary conditions.

3.2.1 Code unit

To solve the system of equations numerically, it is usually convenient to recast them by using dimensionless quantities. In the ExaHyPE2 implementation, we adopt the so-called supercomoving coordinates, which are used in other simulation codes such as RAMSES (Teyssier, 2002).

The original formulation of this coordinate system could be found in Martel and Shapiro, 1998. Its idea is to apply the following rescaling of the variables:

$$\mathrm{d}\tilde{t} \equiv H_0 \frac{\mathrm{d}t}{a^2},$$

$$\begin{aligned}
\tilde{x} &\equiv \frac{1}{a}\frac{x}{L}, \\
\tilde{\rho} &\equiv a^3\frac{\rho(\boldsymbol{x},t)}{\Omega_{\rm m0}\rho_c} \quad = \quad \frac{\rho(\boldsymbol{x},t)}{\bar{\rho}_{\rm m}(t)}, \\
\tilde{p} &\equiv a^5\frac{p}{\Omega_{\rm m0}\rho_c H_0^2 L^2}, \\
\tilde{\boldsymbol{u}} &= a\frac{\boldsymbol{u}}{H_0 L}.
\end{aligned} \tag{3.67}$$

Here $\rho_c, \bar{\rho}_{\rm m}(t)$ are respectively the critical density today and mean density of matter at time t; L is the comoving size of unit code length; dt, x and $\boldsymbol{u}$ denote, respectively, the (physical) time interval, physical coordinate and peculiar velocity. We use the quantities with a tilde in our code, we therefore call them code units in the following context.

The supercomoving coordinate system factors out most of the effect from the Hubble expansion, and thus allows us to implement the original fluid equations (3.16)-(3.19) in a static space with just minor changes. For the special case $\gamma = 5/3$, the only change of the fluid equations is a re-calibration of the gravity term in equation (3.17), which now needs to be derived from the following code-unit Poisson equation:

$$\tilde{\nabla}^2\tilde{\Phi} = \frac{3}{2}\Omega_{\rm m0}a(\tilde{\rho} - 1), \tag{3.68}$$

where $\tilde{\Phi}$ is the Newtonian potential in code unit

$$\tilde{\Phi} = \frac{a^2\Phi}{L^2H_0^2}. \tag{3.69}$$

Solving equation (3.68) under spherical symmetry gives us the following solution of the Newtonian gravitational force $\tilde{g} \equiv -{\rm d}\tilde{\Phi}/{\rm d}\tilde{r}$ (again, in code unit):

$$\tilde{g} = -\frac{3}{2}\Omega_{\rm m0}a\frac{1}{\tilde{r}^2}\int_0^{\tilde{r}}\left[\tilde{\rho}(\tilde{r}') - 1\right]\tilde{r}'^2 d\tilde{r}' \equiv -\frac{3}{2}\Omega_{\rm m0}a\frac{1}{\tilde{r}^2}\frac{\delta\tilde{M}(<\tilde{r})}{4\pi}, \tag{3.70}$$

where we have defined $\delta\tilde{M}(<\tilde{r})$ to be the total "mass perturbation" within radius $\tilde{r}$, i.e., the difference between the total mass therein and the mass in the same region were the density there equal to $\bar{\rho}_{\rm m}$. For other fluid equations, we only need to replace physical quantities with code quantities directly. For cases $\gamma \neq 5/3$, extra terms are needed for supercomoving coordinates (although they are straightforward to derive), which we do not cover here.

The generalisation to calculate the modified gravitational force in the SST model is straightforward: we multiply the fifth-force-to-Newtonian-gravity ratio ξ given in equation (3.48) to equation (3.70) directly to obtain the fifth force in the SST model. Most terms in equation (3.48) are constants or time-dependent functions, and the only term that needs to be rewritten in the code unit is

$$\frac{1+\Delta}{y^3}\tau^2 - 1 = \frac{\frac{4\pi}{3}\rho_i r_i^3\,(1+\Delta)}{\frac{4\pi}{3}\rho_i r^3}\tau^2 - 1 = \frac{m(r_i,t_i)}{\frac{4\pi}{3}\rho_i r^3}\tau^2 - 1, \tag{3.71}$$

where we recall that r_i is the initial radius of the fluid element located at r at time t, and $m\,(r_i,t_i)$ is the total mass enclosed within r_i at the initial time t_i. As no shell crossing happens during the evolution, the mass within the radius of this fluid element remains the same, which means:

$$\frac{1+\Delta}{y^3}\tau^2 - 1 = \frac{m(<r,t)}{\frac{4\pi}{3}\rho_i r^3}\tau^2 - 1, \tag{3.72}$$

where $m(<r,t)$ denotes the total mass enclosed in radius r at time $t > t_i$. In our code implementation, the mass is calculated by counting volumes (see Section 3.2.2 below), and thus $m(<r,t) = \sum_{r_k\le r}\rho_k(t)\ell_k^3$, where the subscript k labels the volumes, ℓ_k is the cubic size of volumes k, and $\rho_k(t)$ is the density (all in physical units). Notice that we have:

$$\frac{\rho_k(t)}{\rho_i}\tau^2 = \frac{\tilde{\rho}_k\rho_{\mathrm{H}}(t)}{\rho_i\tau^{-2}} = \frac{\tilde{\rho}_k\rho_{\mathrm{H}}(t)}{\rho_{\mathrm{H}}(t)} = \tilde{\rho}_k \tag{3.73}$$

in the Einstein-de Sitter universe. Putting equation (3.73) back to equation (3.72), we get

$$\frac{1+\Delta}{y^3}\tau^2 - 1 = \frac{\sum_{r_k<r}\tilde{\rho}_k\ell_k^3}{\frac{4\pi}{3}r^3} - 1 = \frac{\sum_{r_k<r}(\tilde{\rho}_k-1)\ell_k^3}{\frac{4\pi}{3}r^3} = \frac{\delta\tilde{M}(<\tilde{r})}{\frac{4\pi}{3}\tilde{r}^3}, \tag{3.74}$$

where in the second equality we have used $\frac{4\pi}{3}r^3 = \sum_{r_k\le r}\ell_k^3$, while in the final equality we have replaced ℓ_k and r with their code-unit expressions, $\tilde{\ell}_k$ and $\tilde{r}$, which

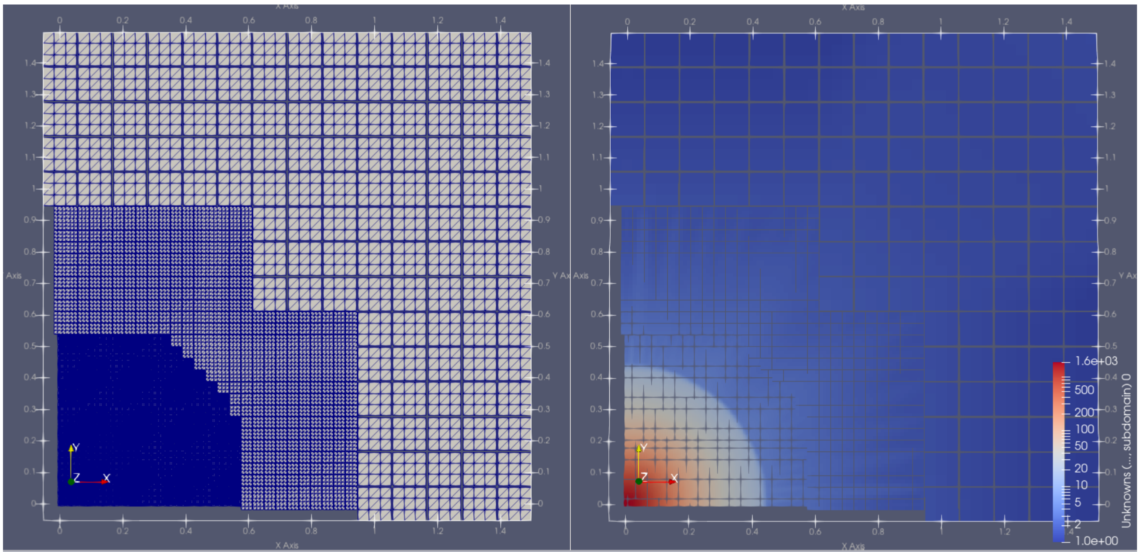

Figure 3.4: *Left Panel:* The adaptive Cartesian grid used in our simulations, with patches and volumes that we describe in chapter 2 therein. The patches with $p = 3$ (i.e., every patch contains 3^3 volumes) are separated from each other in the visualisation with gaps for clarity. Three levels of the grid are shown here. Only one-quarter of x-y plane taken from a slice of the simulation box perpendicular to the z-axis is plotted. The diagonal lines are visualisation artefacts as we use cubic finite volumes. The refinement transitions are conservative, i.e., they are slightly larger than the resolution transitions imposed by the refinement strategy. *Right Panel:* The density field (in code unit) on the same slice for a snapshot during a simulation. Some fluctuations of the density field could be seen out of the central peak, as we discuss them in Section 3.2.3.

does not change the ratio ℓ_k^3/r^3, and used $\delta\tilde{M}(<\tilde{r}) \equiv \sum_{\tilde{r}_k<\tilde{r}} (\tilde{\rho}_k - 1)\,\tilde{\ell}_k^3$. Equation (3.74) is the final code expression that we use in our simulation.

3.2.2 Simulation setups

In the simulations, we implemented the original conservation form of the (gravity-driven) Euler equations in the code unit:

$$\begin{aligned}
\frac{\partial\tilde{\rho}}{\partial\tilde{t}} + \nabla\cdot\tilde{\boldsymbol{j}} &= 0, && (3.75)\\
\frac{\partial\tilde{\boldsymbol{j}}}{\partial\tilde{t}} + \nabla\cdot\left(\frac{1}{\tilde{\rho}}\tilde{\boldsymbol{j}}\otimes\tilde{\boldsymbol{j}} + \tilde{p}\mathbf{I}\right) &= \tilde{\boldsymbol{f}}, && (3.76)\\
\frac{\partial\tilde{E}}{\partial\tilde{t}} + \nabla\cdot\left(\frac{1}{\tilde{\rho}}\tilde{\boldsymbol{j}}(\tilde{E}+\tilde{p})\right) &= \frac{1}{\tilde{\rho}}\tilde{\boldsymbol{j}}\cdot\tilde{\boldsymbol{f}}, && (3.77)
\end{aligned}$$

where $\tilde{\rho}$, $\tilde{\boldsymbol{j}}$, $\tilde{E}$, $\tilde{p}$ represent the density of mass, momentum, energy and pressure in code unit respectively, $\tilde{\boldsymbol{f}} = \tilde{\rho}\tilde{\boldsymbol{g}}$ is the force density with $\tilde{\boldsymbol{g}}$ the gravitational acceleration, which is proportional to $\delta\tilde{M}(<\tilde{r})/\tilde{r}^2$. We consequently obtain $\vec{Q} = (\tilde{\rho}, \tilde{\boldsymbol{j}}, \tilde{\boldsymbol{E}})$ in equation (2.1).

All simulations we presented in this paper use the same grid setup on a cubic box $[-1.5, 1.5]^3$. The maximum refinement level within the tree formalism is 3, corresponding to a resolution of 243^3 patches on the finest level. Every patch contains 27 volumes again ($p = 3$). We coarsen this mesh once at a distance of 0.5 (in code units) away from the origin and coarsen it once more at 0.7. Figure 3.4 illustrates the AMR refinement pattern we used for the simulation. The exact refinement pattern is chosen such that it covers the refinement radii. The safety parameter (CFL ratio) we use in equation (2.17) is $C = 0.3$.

Mass Integration

Most of the terms in equations (3.75) - (3.77) can be implemented in ExaHyPE2 directly as part of the Rusanov scheme on Cartesian meshes we describe above because they are all localised variables, i.e. follow up the updated pattern of any Finite Volume scheme. However, the gravitational force

$$|\tilde{\boldsymbol{f}}| = \tilde{\rho}|\tilde{\boldsymbol{g}}| = \frac{3}{2}\Omega_m \tilde{\rho} a \frac{\delta\tilde{M}(<\tilde{r})}{4\pi\tilde{r}^2}(1+\xi), \tag{3.78}$$

is not localised as we will need the total perturbed mass within radius $\tilde{r}$. To get $\delta\tilde{M}(<\tilde{r})$, we construct a mass array $\{\delta\tilde{m}_i\}_{0\leq i\leq i_{\max}}$ which stores the total perturbed mass values within radii $\{\tilde{r}_i\}_{0\leq i\leq i_{\max}}$. Here $\tilde{r}_{\max} = \tilde{r}_{i=i_{\max}}$ is chosen to be the radius of the largest sphere in the simulation box: half of the domain length. The values of $\delta\tilde{m}_i$ are calculated by accumulating the mass in all volumes that are within $\tilde{r}_i$ per time step (in a parallel computing setup, every thread accumulates their own $\delta\tilde{m}_i$, and we reduce them in the finishing stage of timestep to derive the global one):

$$\delta\tilde{m}_i(<\tilde{r}_i, t) = \sum_{\tilde{r}_k\leq\tilde{r}_i} \left[\tilde{\rho}_k(t) - 1\right]\tilde{\ell}_k^3, \tag{3.79}$$

where $\tilde{\ell}_k$ is the size of the accumulating volume. The plain summation is consistent with our choice of piece-wise constant Finite Volumes. During the subsequent time step, we apply the following interpolation rule per volume according to its radius $\tilde{r}$ for the required perturbed mass:

$$\delta\tilde{M}(<\tilde{r}) = \begin{cases} \delta\tilde{m}_0\tilde{r}^3/\tilde{r}_0^3, & \tilde{r} \leq \tilde{r}_0 \\ \delta\tilde{m}_i\left(\frac{\tilde{r}_{i+1}-\tilde{r}}{\tilde{r}_{i+1}-\tilde{r}_i}\right) + \delta\tilde{m}_{i+1}\left(\frac{\tilde{r}-\tilde{r}_i}{\tilde{r}_{i+1}-\tilde{r}_i}\right), & \tilde{r}_i < r \leq \tilde{r}_{i+1} \\ \delta\tilde{m}_{\max} + \frac{4\pi}{3}\tilde{\rho}(\tilde{r}_{\max})\left(\tilde{r}^3 - \tilde{r}_{\max}^3\right). & \tilde{r} > \tilde{r}_{\max} \end{cases} \tag{3.80}$$

The perturbed masses for volumes outside $\tilde{r}_{\max}$ are approximated by assuming that the density there is equal to that at $\tilde{r}_{\max}$. During our simulations, the densities in those volumes depart little from unity and thus contributes little to the total perturbed mass. This approximation is therefore acceptable. More accurate schemes could be used in future simulations, such as using a scheme of density interpolation that can extend to the furthest corner of the simulation box. Within $\tilde{r}_{\max}$, on the other hand, the accuracy of this interpolation rule depends on the size and arrangement of the sample array $\{\delta\tilde{m}_i\}$. In our simulations, we use a sample array size of 200, and keep our sample radii $\{\tilde{r}_i\}$ invariant over time.

3.2.3 Initial and Boundary Condition

The simulations shown in this chapter start at scale factor $a_i = 0.001$, and end around $a \approx 0.3$. The simulation domain is initially filled with collisional cold gas of $\gamma = 5/3$ in critical density (which is unity in code units). Our overdense seed, the spherical tophat, is placed at the origin and is set to have a radius $\tilde{R}_i = 0.05$ and total perturbed mass $\delta\tilde{M}_i = 0.15$.

Initial Condition

The treatment of the initial conditions of the pressure, density and velocity is subtle. Although we should expect a pressureless infall for most regions in the simulation box at the beginning, we can not set a zero initial pressure numerically. Likewise, although it seems to be quite natural to set a zero initial velocity profile within our comoving coordinate system, we can not do this in our implementation, either. Both of these would lead to negative pressure in the first time step. This is because in this step the energy equation 3.77, does not update the local energy given the zero momentum (i.e., both the flux and the source terms are zero in this equation). On the other hand, the momentum itself is updated normally according to equation (3.76) as its source term (the force density) is nonzero. Since we calculate the pressure using:

$$\tilde{p} = (\gamma - 1)\left(\tilde{E} - \frac{1}{2}\tilde{\mathbf{j}}^2/\tilde{\rho}\right), \tag{3.81}$$

the fact that $\tilde{\boldsymbol{j}}$ is updated (mostly increased in magnitude) while $\tilde{E}$ is not during the first step can cause an accidental and unphysical drop of pressure at the end of this timestep, and frequently (for zero initial pressure, it is always) the pressure turns to be negative where gravity is strong, i.e., near the centre. This issue would be worse if we put a point mass as the overdense seed at the centre, like the one in RAMSES (Teyssier, 2002) because it leads to an extremely large magnitude of the gravity force in the adjacent volumes of the point mass.

To address this negative pressure issue, our solution is three-fold. Firstly, we stick to using a tophat overdensity rather than a point mass as our seed, though it harms the solution partially (see the section for results below). A tophat initial profile smooths the gravity field and reduces the magnitude of potential negative pressure. Secondly, we set a very small but non-zero value for the pressure initially: it makes the system more robust to the pressure drop in the first time step, and can quickly converge to the correct pressureless solution outside the shock later in the simulation. Finally, we introduce a pre-set initial velocity profile. We assume our momentum field has evolved a small period of (physical) time before the simulation begins, according to the initial gravity field:

$$\tilde{\boldsymbol{j}}_i = \Delta_t \tilde{\boldsymbol{g}}_i, \tag{3.82}$$

such that the energy can get updated as well. These adaptions successfully solve the initial negative pressure issue without the explicit construction of consistent initial conditions which does not yield unphysical solutions. The freedom of adjusting our initial conditions without harming the final self-similarity is expected given the convergence of the solution (Alard, 2020), and we have explicitly checked that it is true for our simulation by tuning our initial pressure.

Boundary Conditions and Geometric Setup

Our setup to simulate spherical collapses requires free inflow boundary conditions. Because we expect $\vec{Q}$ to be almost stationary in comoving coordinates (or approaching the Hubble flow physically) as we move away from the centre of the computational domain, homogeneous Neumann boundary conditions can yield the free inflow as long as the computational domain is sufficiently large. However, such a large domain is computationally inefficient or even unfeasible, and it is also not clear whether 'large' is well-defined in an evolving system: the shock propagates outwards towards the border over time, thus making it a challenge to use homogeneous Neumann boundary conditions throughout the entire evolution. We, therefore, use the following hybrid scheme:

$$\vec{Q}_{\text{out}} = \begin{cases} \vec{Q}_{\text{in}}, & \tilde{\rho}_{\text{in}} < 1 \qquad (3.83) \\ \vec{Q}_{\text{in}} + \tilde{\ell}_{\text{in}}\, \boldsymbol{n} \cdot \lim_{\boldsymbol{x} \to \partial\Omega} \boldsymbol{\nabla}^{(1)} \vec{Q}(\boldsymbol{x}), & \text{otherwise}, \qquad (3.84) \end{cases}$$

where $\vec{Q}_{\text{in}}$ and $\vec{Q}_{\text{out}}$ denote, respectively, the solution vectors in the volumes on the inner and outer sides of the boundary (see Fig. 2.3). As we introduce above in section 2.3, The boundary conditions in ExaHyPE2 for finite volume solver are implemented by specifying how the quantities in ghost volumes out of the boundary $\vec{Q}_{\text{out}}$ are calculated from ones in their direct neighbours within the domain $\vec{Q}_{\text{in}}$. In most times, we use the extrapolating boundary condition (3.84), where the superscript $^{(1)}$ means we use the first-order approximation of the gradient $\boldsymbol{\nabla}\vec{Q}$ at $\boldsymbol{x}$ approaching the domain boundary $\partial\Omega$, multiplied by the distance between the two volumes, $\tilde{\ell}_{\text{in}}$.

The different behaviours of these two types of boundary conditions are illustrated in Fig. 3.5. The linearly extrapolated boundary condition is more accurate than the homogeneous Neumann one specified by equation (3.83), but it underestimates the momentum inflow from beyond the boundary. As a result, the code-unit density at

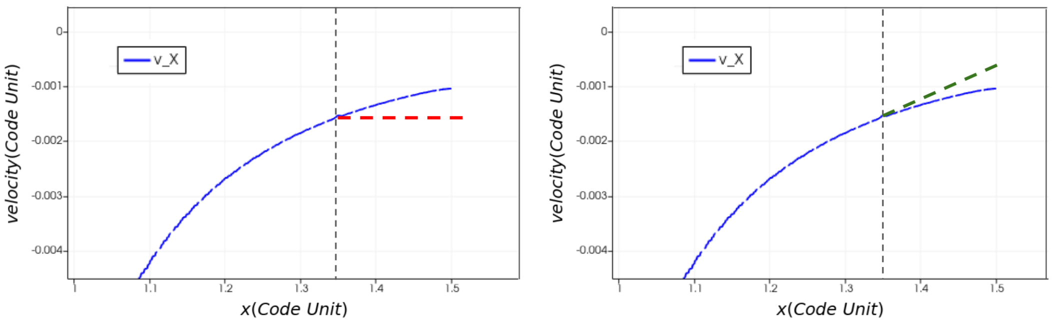

Figure 3.5: The different ways to set up the boundary conditions in the velocity field at the boundary of the simulation domain, indicated by the vertical dashed line. In both panels, the blue curve illustrates a physical velocity profile along the x direction, which has a nonzero gradient at the boundary. *Left panel:* the homogeneous Neumann boundary condition specified by equation (3.83), where the velocity field outside the boundary (i.e., to the right of the vertical line) is assumed to be a constant equal to the velocity value just inside the boundary (the red dashed line). In this case, the inflow from beyond the boundary is overestimated and thus harms the quality of the boundary. *Right panel:* The first-order extrapolated boundary condition corresponding to equation (3.84), as indicated by the green dashed line. Its prediction of inflow is more accurate than the Neumann case but is underestimated. We combine these two boundary conditions in our simulations depending on the local density at the boundary.

the boundary, $\tilde{\rho}_{\text{in}}$, will drop to under unity later in the evolution: this is unphysical because the density everywhere in this collapse scenario should be above the critical density. Whenever this happens, we switch to the homogeneous Neumann boundary condition (3.83). The latter usually overestimates the inflow, and thus can provide some 'compensation'. After the density $\tilde{\rho}_{\text{in}}$ increases back to above unity, we continue using the extrapolating boundary condition again.

The Finite Volume scheme uses normal boundary conditions where the normal is axis-aligned. However, our solution is spherical-symmetric. The boundary condition's normal alignment thus is erroneous. Even with equation (3.83) - (3.84), we have to ensure that the domain remains sufficiently large compared to the area of interest, such that this misalignment becomes negligible. Our box size is chosen based on this consideration. Though the shock only reaches a radius of around 0.5 code units in the final stage of the simulation, we set the radius of the box to be as large as 1.5 units to suppress any boundary pollution. We do not enlarge our box any further as it consumes too much computing resources in regions of little interest. Despite the large computational domain, the finite domain size still limits the maximum simulation time up to which our results are not distorted significantly by the tangential boundary errors, as the solution's steep gradient moves towards the domain boundary.

Similar arguments hold along resolution transitions. As we interpolate linearly along the resolution boundary, our solutions do not follow exact spherical symmetry: the mesh and its resolution transitions should be spherical, and we should interpolate linearly along a spherical transition. Yet, our grid is Cartesian. This "misalignment" results in fluctuations or finger patterns (Fig. 3.4, right panel). Our code has two ingredients to mitigate the resulting error: on the one hand, we use 2:1 balancing (Sundar, Sampath, and Biros, 2008), since a more aggressive resolution change would amplify any error. On the other hand, we ensure that the "first" (finest to second finest) resolution transition is sufficiently far away from the region of interest, i.e. the shock. In return, this implies that the maximum runtime yielding physically admissible results is bounded further, as long as we disable adaptive mesh refinement—a technique which is intrinsically limited, as the area of interest expands and thus eventually yields a regular grid with excessive memory footprint. This numerical error becomes apparent once the shock approaches the resolution transition boundary. In our simulation, we extend the first transition boundary to about 0.6 code units. This is slightly larger than the final 0.5 code units which the shock reaches within code time 60 (corresponds to $a \approx 0.3$). At that time, the shock approaches the transition boundary and thus the numerical error starts to pollute the solutions. As we will see in the result section below, the simulation already reaches the numerical convergence before $a \approx 0.02$, so the current setup provides sufficient time for us to test the code in the stable evolution stage.

3.3 Simulation Tests

In this section, we report the simulation results of spherical collapse scenarios in different gravity models using our application. To make comparison to the theoretical predictions we got in Section 3.1, we will also show results that are rescaled

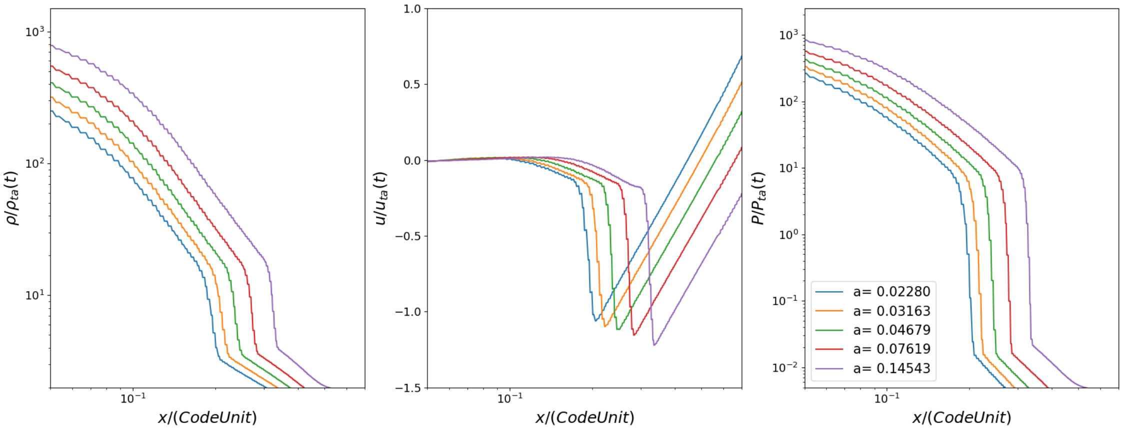

Figure 3.6: The rescaled density, velocity and pressure profiles for the spherical collapse in an Einstein-de Sitter universe, plotted against the radius from the centre in code units. Five snapshots of the system at $a \approx 0.022, 0.031, 0.047, 0.076, 0.145$ are shown in different colours as indicated by the legends. An outward-propagating shock is clearly visible in all three panels. The curves are sampled over the positive direction of the x axis, but we have checked that for all the simulations we report in this section the solution only has a very weak dependence on the direction along which we extract it from the simulation domain, see Appendix A.

following equations (3.21) - (3.24), after we restored the quantities in the physical unit using equation (3.67).

3.3.1 Einstein-de Sitter universe

We first show the simulation results in the Einstein-de Sitter universe. Since gravity is standard, we can use equation (3.15) as the scaling radius. The rescaled profiles of physics quantities are plotted over the radius coordinates in the code unit (supercomoving coordinates), in Fig. 3.6. We illustrate five snapshots of the system (at scale factor $a \approx 0.022$, 0.031, 0.047, 0.076, 0.145) from the late part of the simulation when the corresponding Δ is relatively small. The system remains in stable evolution before the numerical issues we reported in the last section pollute the solution. A clear outward-propagating shock can be seen in the figure.

The same profiles of quantities are plotted again, but now against the rescaled radial coordinates λ, in Fig. 3.7. The theoretical self-similar lines from Section 3.1 are shown as black dashed lines for comparison. We can see a clear self-similarity here, as the rescaled simulated quantities have converged during the time period considered, when the scale factor a increases by a factor of seven. The coloured vertical lines in the figures are the positions of the tophat edge at the time of the corresponding snapshots, within which the density and pressure solutions deviate from the self-similar solution and flatten: this is expected as the gas within the tophat does not experience the full gravity from the mass perturbation anymore. The radius of this edge is shrinking in the rescaled plots over time because the turnaround radius that is used to define λ increases as time evolves.

The rescaled solutions agree with the theoretical predictions quite well, especially for the preshock solutions of the density and velocity. Yet, there are some deviations from the self-similar solution, notably a shift of the shock position. Because of this, the infall velocity of the gas just outside the shock is lower than the theoretical prediction. This is a common numerical artefact caused by volumes with finite widths, which cannot exactly resolve the infinitesimally thin shock. We have checked that the agreement with the self-similar prediction improves as we use finer volumes. A detailed convergence study is beyond the scope here.

Another factor that may have contributed to the difference between theory and simulation is that the theoretical solution here is obtained under the assumption of $\Delta \ll 1$, and this is not well satisfied in the simulations. The different shells of gas have different initial radii r_i and corresponding values of Δ, with the outer shells having larger r_i and therefore smaller Δ, and vice versa. The outer shells also collapse to the shock at a later time. We note that the outer shells that collapse to the shock at later stages of the simulations usually have $\Delta \approx 0.1$, and the inner shells have even larger Δ. The difference between these values of $\Delta \ll 1$ might affect the accuracy of the simulation results. This claim is supported by the time convergence of the profiles in Figure 3.7 toward the self-similar solutions. However, it is not clear to what extent letting the simulation run for longer, so that shells with ever larger r_i will fall to the shock, helps here, since some of the inaccuracy of the simulation results is due to numerical dissipation. Additionally, as we explained in Section 3.2.3, the maximum runtime yielding physically admissible results is bounded, and simulations after a longer time will begin to depart from the self-similar solution generally.

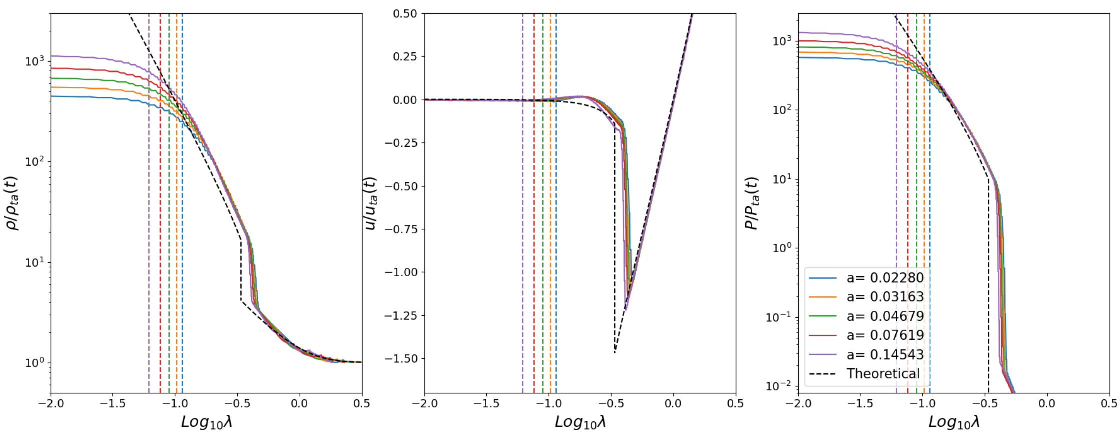

Figure 3.7: The rescaled density, velocity and pressure profiles from the same simulation of the Einstein-de Sitter model, plotted against the rescaled radial coordinate, λ. The self-similar theoretical prediction (Bertschinger, 1985) is shown as black dashed lines. The vertical dashed lines with colours indicate the locations of the tophat edge at the same five times as shown in Fig. 3.6, and the numerical solutions depart from the self-similar prediction within it. This location is moving inwards as the rescaling radius r_{ta} increases over time. Convergence over time to the theoretical solution can be observed in the plots.

We next study the effect of the hybrid boundary condition scheme introduced in Section 3.2.3. Figure 3.8 gives the tail part of the density profiles of three simulations which are identical except for the implementations of the boundary conditions. The three panels correspond to the three types of boundary conditions mentioned above, respectively homogeneous Neumann (outflow), pure linear extrapolation and the hybrid scheme. A clear abnormal uprising of density near the boundary can be seen in the homogeneous case (the first panel), as it overestimates the inflow from beyond the boundary. This effect would "propagate" inwards and eventually pollute the solution, making it unstable. On the other hand, the density drops to under unity (or the critical density in physical units) when we use the extrapolated boundary condition (the second panel), leading to a negative density later in the simulation. By using the hybrid scheme (the third panel), we manage to keep a relatively stable and smooth density evolution near the boundary throughout the simulation.

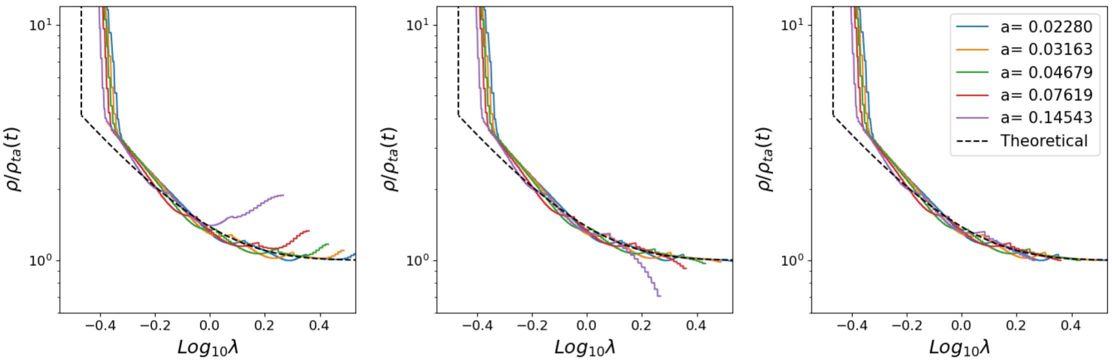

Figure 3.8: The tail parts of the rescaled density profiles from three EdS simulations which implement different boundary conditions but are otherwise identical. The three panels, from left to right, show the results from using homogeneous Neumann (outflow), pure extrapolated and hybrid boundary conditions, respectively. Only in the hybrid case does the profile near the boundary remain stable and consistent with the theoretical prediction (the dashed line). The other two cases either overestimates or underestimates the density near the boundary, leading to an eventual crash of the simulation when the error near the boundary propagates into the central region of the simulation domain (see the purple and red solid lines in the left two panels).

3.3.2 The SST model

In this subsection, we report the simulation results of the SST model introduced in Sections 3.1.1 and 3.1.2, with $\beta = 1.0$ and various values of the screening parameter ζ. For a clear comparison with the standard gravity, we first rescale our modified gravity results using the same turnaround radius formulation equation (3.15), following what we did first in Section 3.1.3. As the rescaling radius is identical in the different gravity models, the differences after the rescaling also represent the difference in the real evolution, thus showing the effects of modified gravity and the screening mechanism.

The results at $a \approx 0.076$ for models with $\beta = 1.0$ and $\zeta = 0,\ 1,\ 5,\ 10,\ 50,\ 100$ are summarised in Figure 3.9. Those results agree with what one should expect for an enhanced gravity force and presence of screening: for the non-screening case ($\zeta = 0$), in which gravity is constantly enhanced in time and space, a stronger shock is observed and it also happens at a larger radius than in EdS. In the other cases, as the screening becomes stronger and stronger (i.e., increasing ζ), the results approach that of standard gravity in an EdS universe.

One may have noticed that we require a bigger ζ to achieve a similar screening

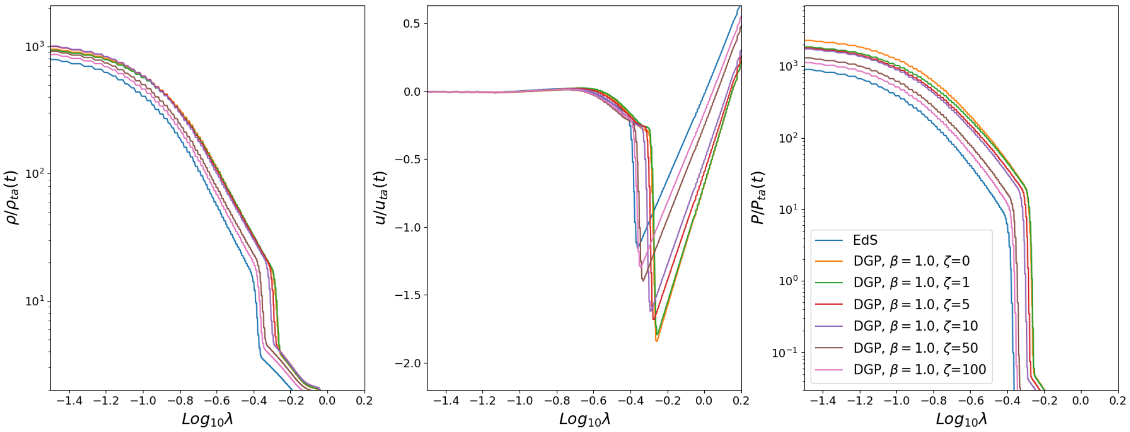

Figure 3.9: The rescaled quantities curves in the SST model introduced in Sections 3.1.1 and 3.1.2, with $\beta = 1.0$ and $\zeta = 0,\ 1,\ 5,\ 10,\ 50,\ 100$, at the time step when $a \approx 0.076$. The profiles in an EdS universe are also illustrated here for a comparison (blue solid lines). The case of $\zeta = 0$ corresponds to a model with a constant (in space and time) enhancement of Newton's constant, while $\zeta > 0$ introduces the Vainshtein screening effect which grows with ζ. It is therefore as expected that the case with a constant enhancement of gravity ($\zeta = 0$) deviates most from the EdS result, and results of other cases lie in between.

effect, compared to Figure 3.1. This is mainly due to the fact that the parameter Δ, which characterises the mean initial overdensity density within some given initial radius r_i, takes different values at the different initial radii covered by a real simulation, while the theoretical profiles are obtained assuming a fixed Δ, e.g., $\Delta = 0.001$. To get rid of the Δ dependence in our results, we use the idea of rescaling using the true turnaround radius as described in Section 3.1.3. The difference is that this time we do not need a "re-rescaling": after we restore the profile quantities in physical units, we find the real turnaround radius directly by its physical meaning, i.e., we locate the radius where the physical velocity crosses zero. This method can be applied to all models including the EdS, which we have checked explicitly to give the same result as in the subsection above. After we located this real turnaround radius for simulations with SST model, we use this value for our rescaling. The result of the same simulations and same timestamp in this new rescaling scheme then are plotted as the solid lines in Fig. 3.10, and their theoretical predictions (as shown in Fig. 3.3) are overplotted as the dashed lines with the same colour scheme. In the figure titles, we have used primes to indicate the quantities calculated using the numerically determined turnaround radius, $r'_{\rm ta}$.

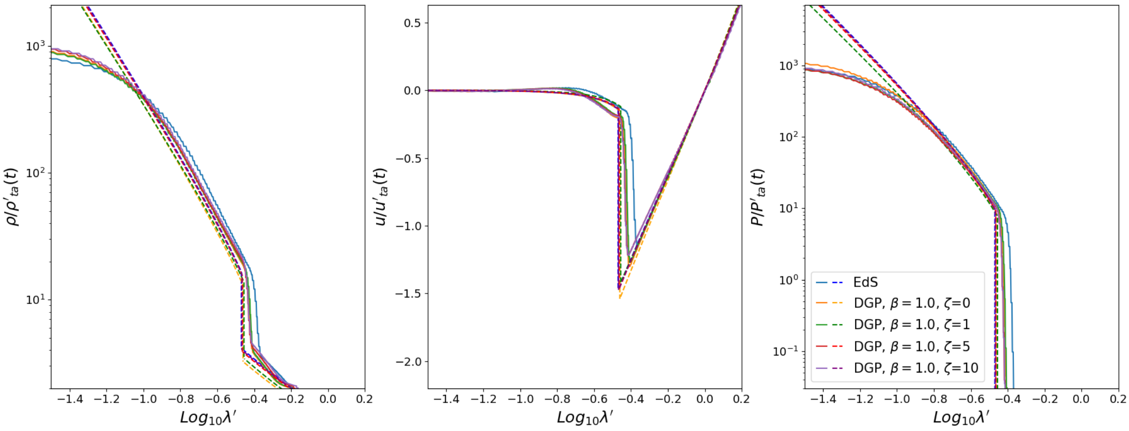

Figure 3.10: The profiles of the same quantities as shown in Fig. 3.9 (solid lines) for the SST model with different parameters (see legends), now rescaled using the real turnaround radius $r'_{\rm ta}$ as described in Section 3.1.3. The quantities with a prime are calculated using this new rescaling radius. We also plot the theoretical self-similar predictions described in Section 3.1.3 for each case, as dashed lines with the corresponding colours.

Just like the theoretical results we got in 3.1.3, the new rescaled solutions are close to that in EdS universe. They are broadly in line with the theoretical predictions as well. The shock in SST model happens at a slightly smaller radius, and the velocity in the gas shell just outside the shock has a bigger magnitude. This result is possibly caused by the fact that the gravitational force is stronger in the SST model, so that the collapse is also stronger and faster. The qualitative trend is also as expected, as the curves for the models with screening are between the ones of EdS and a constant enhancement of Newton's constant ($\zeta = 0$). Given that the real physical evolutions of these models in the simulation are quite different (cf. Fig. 3.9), these results demonstrate the reliability of ExaHyPE2 engine to carry out both standard and modified gravity simulations, and support the idea that self-similarity can be found (at least as a very good approximation) in more general gravity models beyond EdS as well.

3.4 Discussion

In this chapter, we have derived self-similar solutions, for the first time, for some special cases of the DGP class of models. We should point out that the existence of self-similar solutions in spherical collapse scenarios is nontrivial: for example, while the EdS model admits a self-similar solution, this is lost if the model includes a cosmological constant. This is even more true for modified gravity models, in which the law of gravity may be modified in complicated time- and spatial-dependent ways. Indeed, we have tried to search for self-similar solutions in several classes of modified gravity theories that feature certain screening mechanisms. Chameleon-type models (Khoury and Weltman, 2004a; Khoury and Weltman, 2004b) do not admit self-similar solutions, because the fifth force there is not only scale dependent but also environment dependent. We have not found self-similar solutions for K-mouflage-type models (Babichev, Deffayet, and Ziour, 2009; Brax, Burrage, and Davis, 2013) either: in this model, the fifth force is given by

$$F = \beta_{\mathrm{K}} \frac{\mathrm{d}\varphi}{\mathrm{d}r}, \tag{3.85}$$

where β_{K} is a parameter describing the coupling strength of the scalar field φ with the matter, which is usually taken as a constant or function of time. The radial gradient $\mathrm{d}\varphi/\mathrm{d}r$ can be schematically obtained by solving

$$K\left(\frac{\mathrm{d}\varphi}{\mathrm{d}r}\right) \propto \beta_{\mathrm{K}} \frac{m(<r)}{r^2} \propto \beta_{\mathrm{K}} \frac{r_{\mathrm{ta}}}{t^2} \frac{M(\lambda)}{\lambda^2}, \tag{3.86}$$

where $K(\cdot)$ is a nonlinear function, and r_{ta} is again the EdS expression of the turnaround radius, equation (3.15). For the fifth force to also respect self-similarity, it should be possible to express it as

$$F = \mathcal{F}(\lambda) \frac{r_{\mathrm{ta}}}{t^2}, \tag{3.87}$$

where $\mathcal{F}$ is a function of λ only. This is satisfied if $K(\cdot)$ is a linear function and β_{K} is a constant, but this simply corresponds to a model with a constant enhancement of Newton's constant, identical to the SST variant with $\zeta = 0$ considered above. For general $K(\cdot)$, one has to require β_{K} to depend on both r_{ta} (and through which also depend on the initial radius r_i and overdensity Δ) and t to satisfy the above condition. Even for the SST model we considered above, demanding a self-similar solution places some constraints on certain details, in particular the requirement that r_c becomes a time-dependent function which grows at the same rate as the horizon size of the EdS universe. The existence of self-similar solutions in specific models offers us a way to test our numerical code for models other than EdS.

The self-similar solutions we obtained for the SST model behave as one would expect for an enhanced gravity with the Vainshtein screening mechanism at work. For example, we see that the shock happens at a larger radius in the SST variant with $\zeta = 0$, and the infall velocity is larger outside the shock, compared with EdS, as a result of a stronger gravitational collapse. For the other SST variants where $\zeta > 0$, the results generally lie between EdS and $\zeta = 0$, indicating a suppressed fifth force and the suppression effect is larger for bigger ζ. It is notable that, despite the substantial differences in the evolutions and solutions of the different gravity models considered, after the (more 'proper') rescaling using the true turnaround radius of individual models, the solutions in the different SST variants are all very close to that in the EdS model with standard gravity (though their agreement is not perfect). We also notice that, after this proper rescaling, the self-similar solutions in the SST models depend very weakly on Δ, as also happens in EdS. Apparently, we should test these observations for other types of gravity models too. if they hold there as well, this is an interesting indication that the properly rescaled solutions in different gravity models are close to each other, which in turn implies that self-similarity should hold approximately, even though not exactly, in generic models. We leave a more detailed exploration of this possibility to future work.

Behind our new physical insights is a new implementation of cosmological hydrodynamical simulations of the spherical collapse scenario for different gravity models, based on ExaHyPE2. We find that the numerical simulations of the same EdS and SST models as introduced above yield good agreements with the theoretical predictions we derived. This thus not only supports our findings on the self-similarity in the considered models, but also serves as a validation of the reliability and correctness of our ExaHyPE2 application.

By comparing our theoretical predictions to the simulation results, we find that, although to a large degree the code is capable of handling the collapse scenarios in different gravity models, there are still some inaccuracies in the current simulation results, in particular at and around the shock. The observed shift and weakening of the shock are likely caused by numerical dissipation, which may be suppressed by increasing the spatial and temporal resolutions. There are several possible ways of doing this. First, we are currently using a simple Finite Volume formalism which employs a generic Riemann solver. This scheme can be further extended to higher-order formalisms, e.g., the fourth-order Finite Difference method introduced in section 2.2.2, in combination with Runge-Kutta scheme. Those schemes are in principle compatible with our scenarios, straightforward to implement, and could work properly to enhance the resolutions, but it remains an open question if these methods are well-suited to capture the steep gradients near the shock. We could also directly increase the resolution of our simulations, but this comes at an additional runtime cost. For the temporal side, we need to check if local time stepping or subcycling helps to reduce the vulnerability of the current explicit time stepping scheme to numerical dissipation. It may also help to use more accurate Riemann solvers, as the current Rusanov solver only ‘reacts’ to the biggest eigenvalue of the system, cf. equation (2.15)-(2.16), so that it does not preserve the characteristics of all five evolving quantities well if they propagate with different wave speeds.

CHAPTER 4

Application: Numerical Relativity of Black Holes

In this chapter, we give a thorough description of our numerical relativity application on ExaHyPE2. The application aims at simulating black hole spacetimes. We assume readers are familiar with the concepts and notations in General Relativity and will not reintroduce them in this chapter. The theoretical foundation of numerical relativity is presented as a review in section 4.1, mainly following the Baumgarte and Shapiro, 2010 and Shibata, 2015, but also incorporating other related literature which we will cite explicitly. We start from the 3+1 foliation of spacetime and then cover other aspects necessary for the evolution of evolving black holes. After that, we outline the structure of our code in section 4.2 and explain various modules we have implemented in the application. We illustrate the results of the simulation tests in section 4.3 and discuss the existing issues in our code in section 4.4. Finally, in section 4.5, we provide a brief overview of the findings around numerical evolution under the general scalar-tensor theory of gravity. The work in section 4.2-4.5 are all completed by the author in this PhD project.

Throughout the chapter, we adopt the $(-,+,+,+)$ metric signature and follow the standard Einstein convention of summing over repeated indices. We are going to use the Latin letters $a, b, ...$ for spacetime indices which run from 0 to 3, following the *abstract index notation*. The tensor components are labelled by Greek letters $\mu, \nu, \ldots$. For the indices running from 1 to 3, we use the middle part of the Latin letters $i, j, k, \ldots$ as usual. Finally, we will use the geometrized unit system in our content where both the gravitational constant and the speed of light are set to unity, $c = G = 1$.

4.1 Theoretical Background

We introduce the basics of numerical relativity and details of preparing initial conditions for black holes in this section. General Relativity treats space and time on equal footing and the Einstein equation describes the interaction between them and matter content, which is elegant, but not suitable for numerical simulation. We need to translate the system into a Cauchy initial value problem, and that is where the "3+1" decomposition comes in and we need to take the hyperbolicity of the equation, which indicates the stability of the system, into consideration. There are several re-formulations of the Einstein equation that are able to yield a stable evolution of black holes, and we pick the CCZ4 (Alic et al., 2012, Conformal and covariant Z4 system) formulation in our application. It integrates the advantages of the generalised-harmonic formalism and the BSSNOK formalism. therefore yields better stability and accuracy. We also describe how we treat the singularity of black holes in this section. The moving puncture method is applied here to "warp" the singularity so that no physical singularity is resolved in both the initial condition and evolution. Given we are evolving a relativity system, we need to evolve the coordinate system as well, because the coordinates carry the freedom of foliation of the spacetime. It is known as the gauge freedom of general relativity. Coordinate evolution is controlled via the gauge condition. The final part of this section is about the calculation and extraction of gravitation wave signals under the assumption that the observer is far afield.

4.1.1 3+1 Spacetime Foliation in Numerical Relativity

We start our journey into numerical relativity with the Einstein field equation:

$$R_{ab} - \frac{1}{2} g_{ab} R = 8\pi T_{ab} \tag{4.1}$$

The cosmological constant term is omitted here because the length scale relevant to the problem considered is stellar-level, where its effect is negligible. We only consider the standard four-dimensional spacetime in this thesis, though higher-dimensional spacetimes are also of great interest. In four-dimensional spacetime, The index therefore runs over 0 to 3. To convert it into an initial value problem for simulation, we need to decompose the spacetime back to "space" and "time" again.

We define a global scalar field $t(x^a)$ on the manifold as our first step. Using this field, we can naturally split the whole spacetime into a collection of hyperspaces $\sum_t$ which satisfy $t|_{\sum_t}$ = constant. The unit normal vector on the hypersurfaces is defined as

$$n_a := -\alpha g^{ab} \nabla_b t, \tag{4.2}$$

where $\alpha := +\sqrt{-1/g^{ab}\nabla_a t \nabla_b t}$, thus $n^a n_a = -1$. We pick the negative sign convention here to make n^a future pointing ($n^a \nabla_a t > 0$). The spatial metric on hypersurface $\sum_t$ is given as

$$\gamma_{ab} = g_{ab} + n_a n_b. \tag{4.3}$$

One can immediately see $\gamma_{ab} n^b = 0$, this spatial metric can help project quantities into their spatial counterpart, as no temporal components would left after the contraction with it, e.g. $n^b \gamma^a_b T_a \equiv 0$.

The next step is to define the time axis along which we integrate. Mathematically speaking, we are looking for a timelike vector field t^a that satisfies $t^a \nabla_a t = 1$. It guarantees that all t^a on one hypersurface will point onto the same next hypersurface. Most importantly, the time axis is not necessarily normal to the hypersurface $\sum_t$, and its normal part may also differ from the unit normal vector n^μ by a scalar factor. Therefore, we could define two important quantities related to gauge choice by projecting the time vector into the normal part and parallel (spatial) part:

$$\alpha = -t^a n_a, \quad \beta^a := t^b \gamma_b^a. \tag{4.4}$$

These so-called lapse function α and shift vector fields β^i play significant roles in numerical relativity as they contain the four degrees of freedom of coordinates during the evolution. Their relations with the time axis are illustrated in Figure 4.1. Good gauge choices help to enhance stability, improve numerical performance and also avoid singularities. Roughly speaking, the lapse function measures how much proper time passes between two adjacent hypersurfaces and the shift vector describes how the corresponding points are moved from their "original" positions, where they should be if the time axis has a zero shift component. We will go back to the gauge quantities with more details in section 4.1.4 below.

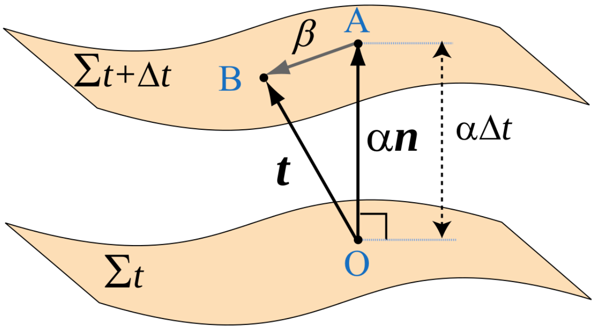

Figure 4.1: The illustration of lapse function and shift vector. Given two neighbouring hypersurfaces labelled as Σ_t and $\Sigma_t + \Delta t$, the four-dimensional time vector OB can be decomposed into OA and AB, which respectively give the proper time α passed for a normal observer at O in the coordinate time Δt, and the coordinate shift between the two hypersurfaces. i.e. points B and O have the same spatial coordinates while the spatial coordinate of A is different by $-\beta^i$. The figure is adapted from Shibata, 2015.

We now have the time axis along which we can do integration, the next step is to construct quantities representing spatial structure. The 3-dimensional covariant derivative on the hypersurfaces is defined as

$$D_a f := \gamma_a^b \nabla_b f, \;\; D_a T_c^b := \gamma_a^d \gamma_e^b \gamma_c^f \nabla_d T_f^e, \tag{4.5}$$

where f and T are general scalar fields and tensors, respectively. This definition guarantees this derivative is compatible with the spatial metric, i.e., $D_a\gamma_{bc} = 0$. The Leibniz rule only holds for this derivative operator when both v^b and w_b are purely spatial

$$D_a\left(v^b w_b\right) = v^b D_a w_b + w_b D_a v^b, \quad \forall\{v^a, w_b | n_a v^a = n^b w_b = 0\}. \tag{4.6}$$

The Christoffel symbol and Riemann tensor associated with the spatial covariant derivative have the same formulations as their 4-dimensional counterpart

$$\Gamma^a_{bc} = \frac{1}{2}\gamma^{ad}\left(\partial_c\gamma_{db} + \partial_b\gamma_{dc} - \partial_d\gamma_{bc}\right), \tag{4.7}$$

$$R^d_{abc} = \partial_b\Gamma^d_{ac} - \partial_a\Gamma^d_{bc} + \Gamma^e_{ac}\Gamma^d_{eb} - \Gamma^e_{bc}\Gamma^d_{ea}. \tag{4.8}$$

We then define the *extrinsic curvature* as the spatial projection of the covariant derivative of the normal vector:

$$K_{ab} := -\gamma^c_a\gamma^d_b\nabla_{(c}n_{d)} = -\gamma^c_a\gamma^d_b\nabla_c n_d = -D_c n_d, \tag{4.9}$$

where $A_{(a}B_{b)} := \frac{1}{2}(A_aB_b + A_bB_a)$ is the symmetrisation parentheses. It has an important physical meaning of how space curvature varies from one hypersurface to the next. It describes how adjacent normal vectors differ from each other (see figure 4.2). The extrinsic curvature is different from the Riemann curvature given above (4.8), which is an *intrinsic curvature* that only describes the curvature inside the hypersurface. If the Riemann curvature is non-zero, a parallel-transported vector may not be parallel to itself anymore after completing a closed loop on the hypersurface, the simplest example being the two-dimensional sphere. The extrinsic and intrinsic curvatures are not necessarily both non-zero (or zero): An infinite-long cylinder surface is an example that has zero intrinsic curvature but non-zero extrinsic curvature. Any parallel-transported vector following a closed loop will be parallel to itself in the end, but the adjacent normal vectors on it differ from each other.

The extrinsic curvature has another good property of being proportional to the

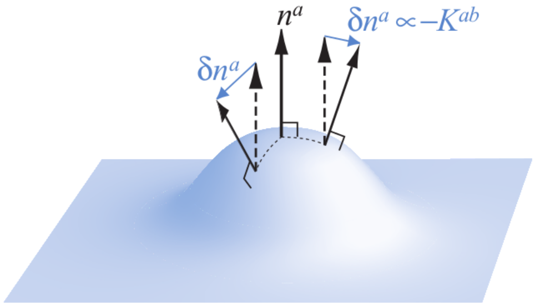

Figure 4.2: The extrinsic curvature is a spatial quantity associated with hypersurfaces, measuring how neighbouring normal vectors differ $\delta n^a \propto -K^{ab}$. It thus shows how a hypersurface distorts and warps through spacetime. The figure is adapted from Baumgarte and Shapiro, 2010.

Lie derivative of the spatial metric, along the normal vector field n^a:

$$K_{ab} = -\frac{1}{2}\mathcal{L}_{\mathbf{n}}\gamma_{ab}. \tag{4.10}$$

The trace of the extrinsic curvature is called mean curvature and is linked to the determinant of the spatial metric

$$K = g^{ab}K_{ab} = \gamma^{ab}K_{ab} = -\frac{1}{2}\gamma^{ab}\mathcal{L}_{\mathbf{n}}\gamma_{ab} = -\frac{1}{2\gamma}\mathcal{L}_{\mathbf{n}}\gamma = -\frac{1}{\gamma^{1/2}}\mathcal{L}_{\mathbf{n}}\gamma^{1/2}, \tag{4.11}$$

where $\gamma := \det\gamma_{ab}$. As $\sqrt{\gamma}\mathrm{d}^3x$ is a volume element of the hypersurface, the mean curvature then measures the fractional change of the three-dimensional volume along the normal direction n^a. One can also see here why we choose a negative sign in the definition of K_{ab} (4.9): a positive mean curvature leads to a contraction of hypersurface volume in this case, which agrees with the common sign convention of curvature.

Starting from the definition of the Lie derivatives one can calculate the so-called *Ricci's equation*:

$$\mathcal{L}_{\mathbf{n}}K_{ab} = n^d n^c \gamma^q_a \gamma^r_b {}^{(4)}R_{drcq} - \frac{1}{\alpha}D_a D_b\alpha - K^c_b K_{ac}. \tag{4.12}$$

It gives the Lie derivative of the extrinsic curvature along n^a. The ${}^{(4)}$ before the Riemann tensor indicates that it is a 4-dimensional quantity. Equations (4.10) and

(4.12) are similar to the dynamic equations of position and velocity in classical mechanics: the evolution of the position (spatial metric) are determined by velocity (extrinsic curvature), while the evolution of the velocity are determined by position (spatial metric, and the gauge). We then transform to the Lie derivative along the time vector t^a from n^a to get the proper "time derivatives" of the evolution system,

$$\mathcal{L}_{\mathbf{t}}\gamma_{ab} = -2\alpha K_{ab} + \mathcal{L}_{\beta}\gamma_{ab}, \tag{4.13}$$

$$\begin{aligned}\mathcal{L}_{\mathbf{t}}K_{ab} = {} & -D_aD_b\alpha + \alpha\left(R_{ab} - 2K_{ac}K^c_b + KK_{ab}\right) - 8\pi\alpha\left(S_{ab} - \frac{1}{2}\gamma_{ab}(S-\rho)\right) \\ & + \mathcal{L}_{\beta}K_{ab},\end{aligned} \tag{4.14}$$

where $\mathcal{L}_{\mathbf{t}} = \alpha\mathcal{L}_{\mathbf{n}} + \mathcal{L}_{\beta}$. The new quantities that appear in the equation (4.14) are the source terms decomposed from the original energy-momentum tensor T_{ab} in the field equation (4.1): $\rho := n_an_bT^{ab}$ is the total energy density measured by a normal observer n^a; $S_{ab} := \gamma^c_a\gamma^d_bT_{cd}$ is spatial stress and $S := S^a_a$ is its trace. There is another source term associated with the momentum density measured by a normal observer: $S_a := -\gamma^b_an^cT_{bc}$.

The tensor equations (4.13) and (4.14), which are component equations of the field equation in essence, give us the prototype of the evolution system. They describe how the space structure evolves among hypersurfaces. On the other hand, the other component equations can be seen as constraints. They have to be satisfied on any hypersurfaces during the evolution, and we utilise them to constrain the initial condition, i.e., how the space structure should be initially. Those constraints are derived from two different projections of the Riemann tensor:

$$\gamma^p_a\gamma^q_b\gamma^r_c\gamma^s_d{}^{(4)}R_{pqrs} = R_{abcd} + K_{ac}K_{bd} - K_{ad}K_{cb}, \tag{4.15}$$

$$\gamma^p_a\gamma^q_b\gamma^r_cn^s{}^{(4)}R_{pqrs} = D_bK_{ac} - D_aK_{bc}, \tag{4.16}$$

which are so-called *Gauss–Codazzi equation* and *Codazzi–Mainardi equation*. We can further simply them by contract twice and once respectively and use the source notation we defined above

$$\begin{aligned} \text{Hamiltonian Constraint} \quad & R + K^2 - K_{ab}K^{ab} &= 16\pi\rho, \qquad (4.17) \\ \text{Momentum Constraint} \quad & D_b K^b_a - D_a K &= 8\pi S_a. \qquad (4.18) \end{aligned}$$

These are important physical constraints we will use frequently later.

As we have already defined the vector field t^a and interpreted it as the time axis, it is natural to construct a coordinate system where t^a is the time basic vector, $t^a = e^a_{(0)} = (1, 0, 0, 0)$. This construction simplifies the system enormously as the Lie derivative along the time vector now reduces to the partial derivative with respect to the t coordinate axis, i.e., $\mathcal{L}_{\mathbf{t}} = \partial_t$. We then construct the remaining three basic vectors, $e^a_{(i)}$, inside a hypersurface which satisfy $n_a e^a_{(i)} = 0$ and $\mathcal{L}_{\mathbf{t}} e^a_{(i)} = 0$. The former makes sure the basic vectors are purely spatial while the latter expands the basic vectors among hypersurfaces. We can immediately find the component expression of n_a from equation (4.4) and $n_a e^a_{(i)} = 0$:

$$n_a = (-\alpha, 0, 0, 0). \tag{4.19}$$

We also have

$$n_a \beta^a = 0 \rightarrow \beta^a = (0, \beta^i). \tag{4.20}$$

The components of the spacetime and spatial metrics can be derived by their definitions and properties in a similar approach:

$$g^{ab} = \begin{pmatrix} -\alpha^{-2} & \alpha^{-2}\beta^i \\ \alpha^{-2}\beta^j & \gamma^{ij} - \alpha^{-2}\beta^i\beta^j \end{pmatrix}, \; g_{ab} = \begin{pmatrix} -\alpha^2 + \beta_l\beta^l & \beta_i \\ \beta_j & \gamma_{ij} \end{pmatrix}. \tag{4.21}$$

$$\gamma^{ab} = \begin{pmatrix} 0 & 0 \\ 0 & \gamma^{ij} \end{pmatrix}, \; \gamma_{ab} = \begin{pmatrix} \beta_l\beta^l & \beta_i \\ \beta_j & \gamma_{ij} \end{pmatrix}. \tag{4.22}$$

We can see all information of spacetime now can be found in the two gauge quantities α, β^i, and the "spatial part of spatial metric", γ_{ij}, under this decomposition. The spacetime line element now is written as

$$ds^2 = -\alpha^2 dt^2 + \gamma_{ij}\left(dx^i + \beta^i dt\right)\left(dx^j + \beta^j dt\right) \tag{4.23}$$

One may already notice that all the quantities that appear in the evolution and constraint equations are purely spatial quantities. As any contravariant spatial tensor satisfies $n_a T^{abc...} = n_b T^{abc...} = ... = 0$, their zeroth components are all zeros. So it is obvious that all information of a contra variant spatial tensor is encoded in its spatial part. It also holds for covariant tensors as the lower metric operator g_{ab} only contains gauges and spatial components of the spatial metric. As a result, if a tensor equation only contains spatial tensors, we can simplify the equation by replacing the spacetime indices with spatial indices directly. It finally leads us to the basic ADM(Arnowitt-Deser-Misner) formulation (Arnowitt, Deser, and Misner, 2008) of numerical relativity, which includes two constraints

$$R + K^2 - K_{ij}K^{ij} = 16\pi\rho, \tag{4.24}$$

$$\nabla_j\left(K^{ij} - \gamma^{ij}K\right) = 8\pi S^i, \tag{4.25}$$

two evolution equations of γ_{ij} and K_{ij}:

$$\partial_t\gamma_{ij} = -2\alpha K_{ij} + \partial_i\beta_j + \partial_j\beta_i, \tag{4.26}$$

$$\begin{aligned}\partial_t K_{ij} = {} & -\nabla_i\nabla_j\alpha + \alpha\left(R_{ij} - 2K_{ik}K^k_j + KK_{ij}\right) - 8\pi\alpha\left(S_{ij} - \frac{1}{2}\gamma_{ij}(S-\rho)\right) \\ & + \beta^k\partial_k K_{ij} + K_{ik}\partial_j\beta^k + K_{kj}\partial_i\beta^k,\end{aligned} \tag{4.27}$$

and two evolution equation of the determinant γ and trace K:

$$\partial_t K = -\nabla^2\alpha + \alpha\left(K_{ij}K^{ij} + 4\pi(\rho + S)\right) + \beta^i\partial_i K, \tag{4.28}$$

$$\partial_t \ln\gamma^{1/2} = -\alpha K + \partial_i\beta^i. \tag{4.29}$$

In the equations above we use the more common symbol with a Latin subscript i: ∇_i to represent the spatial part of the covariant derivative on the hypersurface (D_a, which is compatible with γ_{ab}), and its compatible metric is also the spatial metric

γ_{ij}. We will still use D_a to represent the four-dimensional formulation of the spatial derivative, to distinguish it from ∇_a. The latter is the covariant derivative of the spacetime metric g_{ab}.

The ADM formulation is already complete and can be integrated over time, but it is weak hyperbolic and thus not suitable for stable numerical implementation. Therefore, improved stable formulations based on the ADM system are needed in research. In section 4.1.3, we will present one important approach to deriving strong hyperbolic formulation through the integration of constraints within systems. Additionally, we will introduce in detail the FOCCZ4 (First-order CCZ4) formulation that we use in our application.

4.1.2 Puncture initial condition

We now have the basic evolution system for numerical simulations, and the next step is to find the initial condition of given physical scenarios. The construction of initial condition is not a trivial thing, as it describes the spatial structure of the initial hypersurface $t = 0$ and we need to solve the full constraint equations (4.24) and (4.25), besides specifying the normal physical parameters (e.g., the masses and velocities of objects).

There are different approaches to solving the constraints on the initial hypersurface according to the astrophysics scenarios at hand, and here we only cover the Bowen–York method which is based on the conformal transverse-traceless (CTT) decomposition (Bowen and York, 1980). This method is popular in black hole simulations and is naturally compatible with the *puncture* approach. The latter is designed to achieve singularity avoidance in black hole spacetimes without a need for domain excision.

Our treatment starts with the conformal decomposition of the spatial metric as

$$\bar{\gamma}_{ij} := \psi^{-4}\gamma_{ij}, \; \bar{\gamma}^{ij} = \psi^{4}\gamma^{ij}. \tag{4.30}$$

Here the *conformal factor* ψ is chosen to be $\psi = \det(\gamma_{ij})^{1/12}$ to guarantee the unity of the determinant of the *conformal metric* $\bar{\gamma}_{ij}$: $\bar{\gamma} = \det(\bar{\gamma}_{ij}) = 1$. One can derive the conformal Christoffel symbol from the definition above:

$$\bar{\Gamma}^i_{jk} = \Gamma^i_{jk} - 2\left(\delta^i_j \bar{\nabla}_k \ln\psi + \delta^i_k \bar{\nabla}_j \ln\psi - \bar{\gamma}_{jk}\bar{\gamma}^{il}\bar{\nabla}_l \ln\psi\right). \tag{4.31}$$

where $\bar{\nabla}_i$ is the covariant derivative compatible with the conformal metric $\bar{\gamma}_{ij}$. The contraction of the conformal Christoffel symbol will be used in section 4.1.4 to motivate the shift gauge condition we adopt.

We then define the traceless part of the extrinsic curvature $A_{ij} := K_{ij} - 1/3\gamma_{ij}K$, where $K = \gamma^{ij}K_{ij}$ is the trace of K_{ij}. The conformal decomposition of it uses a different scaling:

$$\bar{A}^{ij} := \psi^{10}A^{ij}, \bar{A}_{ij} = \psi^2 A_{ij}. \tag{4.32}$$

This scaling makes the divergence of A^{ij} and $\bar{A}^{ij}$ consistent: $\nabla_i A^{ij}$ vanishes if and only if $\bar{\nabla}_i A^{ij}$ vanishes. We then put those definitions back into the Hamiltonian and momentum constraints to get their conformal versions

$$8\bar{\nabla}^2\psi - \psi\bar{R} - \frac{2}{3}\psi^5 K^2 + \psi^{-7}\bar{A}_{ij}\bar{A}^{ij} = -16\pi\psi^5\rho, \tag{4.33}$$

$$\bar{\nabla}_j\bar{A}^{ij} - \frac{2}{3}\psi^6\bar{\gamma}^{ij}\bar{\nabla}_j K = 8\pi\psi^{10}S^i. \tag{4.34}$$

where $\bar{R}$ are the Ricci scalar associated with conformal metric $\bar{\gamma}_{ij}$. Its formulation can be calculated from its definition and relation (4.31). $\bar{A}^{ij}$ is a symmetric and traceless tensor, and can be further decomposed into a divergence-less part and a symmetric, traceless gradient of a vector (potential):

$$\bar{A}^{ij} = \bar{A}^{ij}_{TT} + \bar{A}^{ij}_L, \tag{4.35}$$

such that

$$\bar{\nabla}_j\bar{A}^{ij}_{TT} = 0,\ \bar{A}^{ij}_L = \bar{\nabla}^i W^j + \bar{\nabla}^j W^i - \frac{2}{3}\bar{\gamma}^{ij}\bar{\nabla}_k W^k := (\bar{L}W)^{ij}, \tag{4.36}$$

where W^i is a vector potential. Then the divergence of the $\bar{A}^{ij}$ turns into

$$\bar{\nabla}_j A^{ij} = \bar{\nabla}_j A^{ij}_L = \bar{\nabla}_j(\bar{L}W)^{ij} = \bar{\nabla}^2 W^i + \frac{1}{3}\bar{\nabla}^i\left(\bar{\nabla}_j W^j\right) + \bar{R}^i_j W^j := \left(\bar{\Delta}_L W\right)^i. \tag{4.37}$$

The geometry property ($\nabla_a \nabla_b v_c - \nabla_b \nabla_a v_c = R^d_{cab} v_d$) of the Riemann tensor is used in the third equality of the equation above. $\bar{L}$ and $\bar{\Delta}_L$ are *vector gradient* and *vector Laplacian* respectively. With the new notation, the momentum constraint becomes

$$\left(\bar{\Delta}_L W\right)^i - \frac{2}{3}\psi^6 \bar{\gamma}^{ij} \bar{\nabla}_j K = 8\pi \psi^{10} S^i. \tag{4.38}$$

If we utilise our freedom of choosing the coordinates and shape of the initial hypersurface in spacetime by assuming maximal slicing $K = 0$ and conformal flatness $\bar{\gamma}_{ij} = \eta_{ij}$, the equation can be simplified to a form that can be solved analytically in the vacuum:

$$\partial^j \partial_j W^i + \frac{1}{3}\partial^i \partial_j W^j = 0. \tag{4.39}$$

Further assuming that the divergence-less part $\bar{A}^{ij}_{TT}$ vanishes, the equation above gives a solution in black hole spacetime:

$$\bar{A}^{ij}_{\mathbf{P}} = \frac{3}{2r^2}\left(P^i n^j + P^j n^i - \left(\eta^{ij} - n^i n^j\right) n^k P_k\right), \tag{4.40}$$

$$\bar{A}^{ij}_{\mathbf{S}} = \frac{6}{r^3} n^{(i} \epsilon^{j)kl} S_k n_l, \tag{4.41}$$

where **P** and **S** represents two components of momentum. This set of solutions is often called *Bowen-York solutions*. Here P^k and S_k represent the linear and angular momentum (spin) of black holes, r is the coordinate distance from the black holes so for a black hole located at the origin it gives $r = \sqrt{x^2 + y^2 + z^2}$ and $n^i = x^i/r$. For those not located there the corresponding quantities are replaced by $n^i = x^i - C^i/r, r = \|x^i - C^i\|$, C^i being the location of the black hole. One may notice the momentum constraint now becomes linear (regarding the W^i, thus also on $\bar{A}^{ij}$), so we can construct $\bar{A}_{ij}$ for multiple black holes by superposition directly:

$$\bar{A}^{ij} = \bar{A}^{ij}_{\mathbf{C}_1\mathbf{P}_1} + \bar{A}^{ij}_{\mathbf{C}_1\mathbf{S}_1} + \bar{A}^{ij}_{\mathbf{C}_2\mathbf{P}_2} + \bar{A}^{ij}_{\mathbf{C}_2\mathbf{S}_2} + ... \tag{4.42}$$

This is the essence of the Bowen-York approach: it solves the momentum constraints analytically and leaves us only the Hamiltonian constraint that needs to be solved numerically, which also gets simplified under the assumptions of maximal slicing and conformal flatness. It is now written as:

$$\bar{\nabla}^2\psi = -\frac{1}{8}\psi^{-7}\bar{A}_{ij}\bar{A}^{ij}. \tag{4.43}$$

The right-hand side can be calculated analytically, but it diverges at the centres of the black holes (singularities). This is actually expected because we are solving for the black holes. As we mentioned before, there are multiple approaches to avoid singularities, including excision: remove the region close to black holes and find a proper internal condition on that boundary, the rest of the spacetime is rather smooth and yields a stable evolution. However, the puncture method (Brandt and Brügmann, 1997) allows us to evolve the whole spacetime and evades the difficulty in the imposition of internal boundaries. This method is, therefore, much easier to implement in practice.

The key idea of the puncture method is to absorb the singularities in analytical terms and leave the smooth part of the constraint for a numerical solution. In this treatment, the conformal metric is assumed to have the form of

$$\psi = \psi_{BH} + u, \;\; \psi_{BH} := 1 + \sum_{i=1}^{N}\frac{m_i}{2r_i}, \tag{4.44}$$

where ψ_{BH} represent the analytical part and u is the residual required numerical treatment. m_i is the so-called puncture mass or bare mass. It is linked to the ADM mass M_i by

$$M_i = m_i\left(1 + u_i + \sum_{i\neq j}\frac{m_j}{2d_{ij}}\right) \tag{4.45}$$

in non-spinning cases. Here M_i is the ADM mass of the ith black hole and u_i is the value of the u field at its location. d_{ij} is the coordinate distance between the ith and jth black holes. In the case of a spinning black hole, the mass is modified according to the Christodoulou formula (Christodoulou, 1970)

$$M_{i,spin}^2 = M_i^2 + \frac{S_i^2}{4M_i^2}. \tag{4.46}$$

Putting the expression (4.44) back into the Hamiltonian constraint, we get

$$\bar{\nabla}^2 u = -\beta(\alpha(1+u)+1)^{-7}, \beta \equiv \frac{1}{8}\alpha^7 \bar{A}_{ij}\bar{A}^{ij} \tag{4.47}$$

This equation is C^∞ anywhere in R^3 and C^2 at the puncture (Brügmann et al., 2008), therefore can be solved by standard numerical treatment.

The current initial condition code in our code is a ported version of the TwoPunctures module from the numerical library EINSTEINTOOLKIT (Löffler et al., 2012), following the numerical methods introduced in Ansorg, Brügmann, and Tichy, 2004. Several coordinate transformations are implemented to transform the unbounded R^3 into a bounded domain, and then the standard 3-dimensional spectral method is used for a high-accuracy and quick-convergence solution.

Notice that we are using different definitions of physics quantities, mainly the conformal scaling, in the initial condition construction and evolution equations (see section 4.1.3). This does not affect the consistency of the code as the two processes are separate in coding practice and are bonded via the ADM quantities γ_{ij} and K_{ij}. We recover the spatial metric and extrinsic curvature before the first time step, and the calculation of other variables that are needed in the evolution system also happens at that time. We will describe the workflow of our code in more detail in section 4.2.

The last piece in the initial condition is the value of gauge quantities α and β^i. Their evolution equations will be covered in section 4.1.4. In black hole simulations, the initial shift β^i is commonly set to be zero, while the choice for the initial lapse is kind of arbitrary and needs empirical insights. Currently, we adopt the formulation of initial lapse used in tests of Dumbser et al., 2018 as

$$\alpha = \frac{1}{2}\left(\frac{1-\frac{1}{2}\left(m_-/r_-\right)-\frac{1}{2}\left(m_+/r_+\right)}{1+\frac{1}{2}\left(m_-/r_-\right)+\frac{1}{2}\left(m_+/r_+\right)}+1\right), \tag{4.48}$$

where r_+ and r_- are coordinate distances of the required point from the two black holes. This is one choice in the optional initial lapse profiles available in the TwoPunctures module.

Before we close this section, let us briefly comment on why the puncture method can help in simulating a spacetime with singularity without excision. One important property of punctured spacetimes is that the spacetime geometry actually becomes flat again when we approach any punctures in space (Brill and Lindquist, 1963). Therefore, the singularity at the punctures can be seen as an asymptotically flat end of the "other" universe, and it is a coordinate singularity rather than a physical singularity. This property is preserved through the evolution under a proper gauge (i.e., the "1+log" condition, see section 4.1.4), as long as the singularities do not hit any grid point themselves. This is a significant advantage of the so-called "moving puncture" method, and this method is proven to be extremely successful in simulating black holes with long-term stability (Brügmann et al., 2008). To further support this, Hannam et al., 2007 examined the puncture method in detail and found that the original $1/r$ singularity of a single static black hole is replaced by a flatter $1/\sqrt{r}$ singularity after the evolution of a short time. The $r \to 0$ singularity approaches a certain areal radius which is within the horizon $r_s = 2M$, rather than another flat end of spacetime. As it is still a coordinate singularity, The physical singularity is again not included in the numerical domain and, thus automatically avoided.

4.1.3 Hyperbolic evolving formulations and FOCCZ4

In section 4.1.1, we already derive the ADM evolution equations (4.26) - (4.27) based on the quantities γ_{ij} and K_{ij}. Analysis has shown that this formulation is only weak hyperbolic and the numerical errors arising in the evolution are very likely to accumulate and finally crash the simulation. Codes in the ADM formulation only yield unstable evolution even in some quite simple scenarios (Baumgarte and Shapiro, 1998b).

Several strategies have been proposed in the literature to tackle this instability. One direction focuses on the hyperbolicity enhancement of the evolution system by introducing auxiliary variables from the constraints (4.24) and (4.25). The Constraints enter the system in a similar manner to the Lagrangian multiplier and those new degrees of freedom affect the characteristic matrices of the system, hence the hyperbolicity (Baumgarte and Shapiro, 2010).

The highly successful BSSNOK formulation in numerical relativity is based on this idea. It involves the momentum constraint in its system by treating the contracted Christofell symbol $\bar{\Gamma}^i$ as an independent variable. $\bar{\Gamma}^i$, which is sometimes also called *conformal connection function*, is actually the contraction of the partial derivative of the contravariant conformal metric $\bar{\gamma}^{ij}$. Therefore, the evolution of this quantity is completely determined by γ_{ij} in the original ADM formulation. The BSSNOK formulation recasts the evolution equation of $\bar{\Gamma}^i$ using the momentum constraint, making this quantity a new degree of freedom which represents the deviation of the system from the momentum constraint.

The Z4 formulation follows a similar approach but is more first-principle-based. It introduces an auxiliary vector field Z^a, whose components represent the four degrees of freedom from the Hamiltonian and momentum constraints. This vector field is added to the Lagrangian density of spacetime as

$$\mathcal{L} = g^{ab}\left[R_{ab} + 2\nabla_a Z_b\right]. \tag{4.49}$$

This new vector field can be seen as a measurement of the (numerical) deviation of the system from the Einstein equations. Following the usual variational approach, this Lagrangian leads to the following new field equations, with damping terms of small coefficients κ_1 and κ_2:

$$R_{ab} - \frac{1}{2}g_{ab}R + \nabla_a Z_b + \nabla_b Z_a - g_{ab}\nabla^c Z_c - \kappa_1[n_a Z_b + n_b Z_a + \kappa_2 g_{ab} n_c Z^c] = 8\pi T_{ab}, \tag{4.50}$$

or with trace reversed:

$$R_{ab} + \nabla_a Z_b + \nabla_b Z_a - \kappa_1[n_a Z_b + n_b Z_a. - (1+\kappa_2)\, g_{ab} n_c Z^c] = 8\pi\left(T_{ab} - \frac{1}{2}g_{ab}T\right). \tag{4.51}$$

These four-dimensional equations can be written in the 3+1 foliation approach we discussed in Section 4.1.1, giving (Bona et al., 2003b):

$$\left(\partial_t - \mathcal{L}_\beta\right)\gamma_{ij} = -2\alpha K_{ij}, \tag{4.52}$$

$$
\begin{aligned}
\left(\partial_t-\mathcal{L}_\beta\right) K_{ij}= & -\nabla_i \nabla_j \alpha+\alpha[R_{ij}+\nabla_i Z_j+\nabla_j Z_i-2 K_i^l K_{lj}+(K-2 \Theta) K_{ij} \\
& -\kappa_1\left(1+\kappa_2\right) \Theta \gamma_{ij}]-8 \pi \alpha\left[S_{ij}-\frac{1}{2}(S-\rho) \gamma_{ij}\right],
\end{aligned} \tag{4.53}
$$

$$
\begin{aligned}
\left(\partial_t-\mathcal{L}_\beta\right) \Theta= & \frac{\alpha}{2}\Big[R+2 \nabla_j Z^j+(K-2 \Theta) K-K^{ij} K_{ij}-2 \frac{Z^j \nabla_j \alpha}{\alpha} \\
& -2 \kappa_1\left(2+\kappa_2\right) \Theta-16 \pi \rho\Big],
\end{aligned} \tag{4.54}
$$

$$
\left(\partial_t-\mathcal{L}_\beta\right) Z_i=\alpha\left[\nabla_j\left(K_i^j-\delta_i^j K\right)+\partial_i \Theta-2 K_i^j Z_j-\Theta \frac{\nabla_j \alpha}{\alpha}-\kappa_1 Z_i-8 \pi S_i\right]. \tag{4.55}
$$

where $\Theta \equiv -n_a Z^a = \alpha Z^0$ and Z^i is the spatial part of $\gamma^a_b Z^b$. The detailed derivation of the above equations can be found in Appendix B.1.1.

We now move forward to the CCZ4, i.e., conformal and covariant Z4 formulation. Similar to the treatment in section 4.1.2, we apply a conformal decomposition on γ_{ij} and write the metric in terms of a conformal factor ϕ and a conformal metric $\tilde{\gamma}_{ij}$, such that

$$
\tilde{\gamma}_{ij} = \phi^2 \gamma_{ij}, \ \phi = [\det(\gamma_{ij})]^{-1/6}. \tag{4.56}
$$

The second equation guarantees that $\tilde{\gamma}_{ij}$ has a unit determinant. We also decompose the extrinsic curvature into its trace part $K = K_{ij}\gamma^{ij}$ and traceless part A_{ij}, but this time the conformal counterpart of A_{ij} is defined using the same scaling as for the metric for consistency:

$$
\tilde{A}_{ij} = \phi^2 A_{ij} = \phi^2 (K_{ij} - \frac{1}{3} K \gamma_{ij}). \tag{4.57}
$$

Similarly to the BSSNOK formulation, we need to treat the contracted Christoffel symbol as an independent variable. But instead of using it directly, we evolve a slightly modified version of it in the CCZ4 formulation, involving the Z field:

$$
\hat{\Gamma}^i \equiv \tilde{\Gamma}^i + 2\tilde{\gamma}^{ij} Z_j, \ \tilde{\Gamma}^i \equiv \tilde{\gamma}^{jk}\tilde{\Gamma}^i_{jk} = \tilde{\gamma}^{ij}\tilde{\gamma}^{kl}\partial_l \tilde{\gamma}_{jk} = -\partial_j \tilde{\gamma}^{ij}. \tag{4.58}
$$

The expression of $\tilde{\Gamma}^i$ is simplified, similar to what we see in the gauge section. Now the degrees of freedom of Z^i, which are from the momentum constraint, are succeeded by $\hat{\Gamma}^i$. Inserting those definitions back into equations (4.52)-(4.55), the full equations of the CCZ4 evolution system are given as (Alic et al., 2012):

$$\partial_t\tilde{\gamma}_{ij} = -2\alpha\tilde{A}^{\mathrm{TF}}_{ij} + 2\tilde{\gamma}_{k(i}\partial_{j)}\beta^k - \frac{2}{3}\tilde{\gamma}_{ij}\partial_k\beta^k + \beta^k\partial_k\tilde{\gamma}_{ij}, \tag{4.59}$$

$$\begin{aligned}\partial_t\tilde{A}_{ij} = \phi^2\left[-\nabla_i\nabla_j\alpha + \alpha\left(R_{ij} + \nabla_i Z_j + \nabla_j Z_i - 8\pi S_{ij}\right)\right]^{\mathrm{TF}} + \alpha\tilde{A}_{ij}(K - 2\Theta)\\ - 2\alpha\tilde{A}_{il}\tilde{A}^l_j + 2\tilde{A}_{k(i}\partial_{j)}\beta^k - \frac{2}{3}\tilde{A}_{ij}\partial_k\beta^k + \beta^k\partial_k\tilde{A}_{ij},\end{aligned} \tag{4.60}$$

$$\partial_t\phi = \frac{1}{3}\alpha\phi K - \frac{1}{3}\phi\partial_k\beta^k + \beta^k\partial_k\phi, \tag{4.61}$$

$$\begin{aligned}\partial_t K = -\nabla^i\nabla_i\alpha + \alpha\left(R + 2\nabla_i Z^i + K^2 - 2\Theta K\right) + \beta^j\partial_j K\\ - 3\alpha\kappa_1\left(1 + \kappa_2\right)\Theta + 4\pi\alpha(S - 3\rho),\end{aligned} \tag{4.62}$$

$$\begin{aligned}\partial_t\Theta = \frac{1}{2}\alpha\left(R + 2\nabla_i Z^i - \tilde{A}_{ij}\tilde{A}^{ij} + \frac{2}{3}K^2 - 2\Theta K\right) - Z^i\partial_i\alpha + \beta^k\partial_k\Theta\\ - \alpha\kappa_1\left(2 + \kappa_2\right)\Theta - 8\pi\alpha\rho,\end{aligned} \tag{4.63}$$

$$\begin{aligned}\partial_t\hat{\Gamma}^i = 2\alpha\left(\tilde{\Gamma}^i_{jk}\tilde{A}^{jk} - 3\tilde{A}^{ij}\frac{\partial_j\phi}{\phi} - \frac{2}{3}\tilde{\gamma}^{ij}\partial_j K\right) + 2\tilde{\gamma}^{ki}\left(\alpha\partial_k\Theta - \Theta\partial_k\alpha - \frac{2}{3}\alpha K Z_k\right)\\ - 2\tilde{A}^{ij}\partial_j\alpha + \tilde{\gamma}^{kl}\partial_k\partial_l\beta^i + \frac{1}{3}\tilde{\gamma}^{ik}\partial_k\partial_l\beta^l + \frac{2}{3}\tilde{\Gamma}^i\partial_k\beta^k - \tilde{\Gamma}^k\partial_k\beta^i\\ + 2\kappa_3\left(\frac{2}{3}\tilde{\gamma}^{ij}Z_j\partial_k\beta^k - \tilde{\gamma}^{jk}Z_j\partial_k\beta^i\right) + \beta^k\partial_k\hat{\Gamma}^i\\ - 2\alpha\kappa_1\tilde{\gamma}^{ij}Z_j - 16\pi\alpha\tilde{\gamma}^{ij}S_j.\end{aligned} \tag{4.64}$$

$$\partial_t\alpha = -2\alpha(K - 2\Theta) + \beta^k\partial_k\alpha, \tag{4.65}$$

$$\partial_t\beta^i = f b^i + \beta^k\partial_k\beta^i, \tag{4.66}$$

$$\partial_t b^i = \partial_t\hat{\Gamma}^i - \beta^k\partial_k\hat{\Gamma}^i + \beta^k\partial_k b^i - \eta b^i. \tag{4.67}$$

The TF index in equations (4.59) and (4.60) means the trace-free part of the quantities, i.e., $R^{\mathrm{TF}}_{ij} = R_{ij} - \frac{1}{3}\gamma_{ij}\gamma^{kl}R_{kl}$. A new parameter κ_3 is introduced above in equation (4.64). In most cases, κ_3 is set to be 1 for a covariance system, while it has to be changed to a smaller value (e.g., 1/2) to achieve a stable evolution in a black-hole system. In the above equations, we also include the gauge conditions (4.65)-(4.67). They are slightly different from the original 1+log slicing (4.103) and the Gamma driver gauge (4.108), by replacing K and $\tilde{\Gamma}^i$ with $K - 2\Theta$ and $\hat{\Gamma}^i$ to include the effect of the vector Z^a during evolution. The detailed derivation of the above equations can be found in Appendix B.1.2.

The original CCZ4 formulation is based on the equations above, and we call this evolution system second order because it requires variables and their first and second spatial derivatives. We can further introduce a set of auxiliary variables to recast the system into a first-order formulation Dumbser et al., 2018, which we call FOCCZ4 below. This first-order formulation is more suitable to the PDE template of ExaHyPEand allows us to tune its hyperbolicity in more detail. The new auxiliary variables are given as:

$$A_i := \partial_i \alpha, \quad B_k^i := \partial_k \beta^i, \quad D_{kij} := \frac{1}{2}\partial_k \tilde{\gamma}_{ij}, \quad P_i := \partial_i \phi. \tag{4.68}$$

Defined as the derivative of quantities, these variables satisfy the natural second-order constraints as

$$\partial_k A_i - \partial_i A_k = 0, \quad \partial_k B_l^i - \partial_l B_k^i = 0, \quad \partial_k D_{lij} - \partial_l D_{kij} = 0, \quad \partial_k P_i - \partial_i P_k = 0. \tag{4.69}$$

Because $\tilde{A}_{ij}$ is traceless $\gamma^{ij}\tilde{A}_{ij} = 0$ and $\tilde{\gamma} = \det(\tilde{\gamma}_{ij}) = 1$, we also have

$$\partial_k \left(\tilde{\gamma}^{ij} \tilde{A}_{ij} \right) = \partial_k \tilde{\gamma}^{ij} \tilde{A}_{ij} + \tilde{\gamma}^{ij} \partial_k \tilde{A}_{ij} = 0, \quad \tilde{\gamma}^{ij} D_{kij} = 0. \tag{4.70}$$

The formulation of the FOCCZ4 evolution system is thus given as the following nine equations for primary variables:

$$\begin{aligned}
\partial_t \tilde{\gamma}_{ij} = {} & \beta^k 2 D_{kij} + \tilde{\gamma}_{ki} B_j^k + \tilde{\gamma}_{kj} B_i^k - \frac{2}{3}\tilde{\gamma}_{ij} B_k^k - 2\alpha \left(\tilde{A}_{ij} - \frac{1}{3}\tilde{\gamma}_{ij} \operatorname{tr} \tilde{A} \right) \\
& - \tau^{-1}(\tilde{\gamma} - 1)\tilde{\gamma}_{ij},
\end{aligned} \tag{4.71}$$

$$\partial_t \alpha = \beta^k A_k - \alpha^2 g(\alpha) \left(K - K_0 - 2c\Theta \right), \tag{4.72}$$

$$\partial_t \beta^i = \beta^k B_k^i + f b^i, \tag{4.73}$$

$$\partial_t \phi = \beta^k P_k + \frac{1}{3}\phi \left(\alpha K - B_l^l \right), \tag{4.74}$$

$$\begin{aligned}
& \partial_t \tilde{A}_{ij} - \beta^k \partial_k \tilde{A}_{ij} - \phi^2 \left[-\nabla_i \nabla_j \alpha + \alpha \left(R_{ij} + \nabla_i Z_j + \nabla_j Z_i \right) \right]^{\mathrm{TF}} \\
& \quad = \tilde{A}_{ki} B_j^k + \tilde{A}_{kj} B_i^k - \frac{2}{3}\tilde{A}_{ij} B_k^k + \alpha \tilde{A}_{ij}(K - 2\Theta c) - 2\alpha \tilde{A}_{il} \tilde{\gamma}^{lm} \tilde{A}_{mj} \\
& \qquad - \tau^{-1} \tilde{\gamma}_{ij} \operatorname{tr} \tilde{A},
\end{aligned} \tag{4.75}$$

$$\begin{aligned}\partial_t K - \beta^k\partial_k K + \nabla^i\nabla_i\alpha - \alpha\left(R + 2\nabla_i Z^i\right)\\ = \alpha K(K - 2\Theta c) - 3\alpha\kappa_1\left(1+\kappa_2\right)\Theta,\end{aligned}\tag{4.76}$$

$$\begin{aligned}\partial_t\Theta - \beta^k\partial_k\Theta - \frac{1}{2}\alpha e^2\left(R + 2\nabla_i Z^i\right)\\ = \frac{1}{2}\alpha e^2\left(\frac{2}{3}K^2 - \tilde{A}_{ij}\tilde{A}^{ij}\right) - \alpha\Theta K c - Z^i A_i - \alpha\kappa_1\left(2+\kappa_2\right)\Theta,\end{aligned}\tag{4.77}$$

$$\begin{aligned}&\partial_t\hat{\Gamma}^i - \beta^k\partial_k\hat{\Gamma}^i - 2\alpha\tilde{\gamma}^{ki}\partial_k\Theta - \tilde{\gamma}^{kl}\partial_{(k}B^i_{l)} - s\frac{1}{3}\tilde{\gamma}^{ik}\partial_{(k}B^l_{l)} - 2\alpha\tilde{\gamma}^{ik}\tilde{\gamma}^{nm}\partial_k\tilde{A}_{nm}\\ &+ \frac{4}{3}\alpha\tilde{\gamma}^{ij}\partial_j K = \frac{2}{3}\tilde{\Gamma}^i B^k_k - \tilde{\Gamma}^k B^i_k + 2\alpha\left(\tilde{\Gamma}^i_{jk}\tilde{A}^{jk} - 3\tilde{A}^{ij}\frac{P_j}{\phi}\right)\\ &- 2\alpha\tilde{\gamma}^{ki}\left(\Theta\frac{A_k}{\alpha} + \frac{2}{3}KZ_k\right) - 2\tilde{A}^{ij}A_j - 4\alpha\tilde{\gamma}^{ik}D_k^{nm}\tilde{A}_{nm}\\ &+ 2\kappa_3\left(\frac{2}{3}\tilde{\gamma}^{ij}Z_j B^k_k - \tilde{\gamma}^{jk}Z_j B^i_k\right) - 2\alpha\kappa_1\tilde{\gamma}^{ij}Z_j,\end{aligned}\tag{4.78}$$

$$\partial_t b^i - \beta^k\partial_k b^i = \partial_t\hat{\Gamma}^i - \beta^k\partial_k\hat{\Gamma}^i - \eta b^i,\tag{4.79}$$

and four equations for the auxiliary variables:

$$\begin{aligned}&\partial_t A_k - \beta^l\partial_l A_k + \alpha^2 g(\alpha)\left(\partial_k K - \partial_k K_0 - 2c\partial_k\Theta\right) + \alpha g(\alpha)\tilde{\gamma}^{nm}\partial_k\tilde{A}_{nm}\\ &\quad = B^l_k A_l - \left[2\alpha g(\alpha) + \alpha^2 g'(\alpha)\right]\left(K - K_0 - 2c\Theta\right)A_k + 2\alpha g(\alpha)D_k^{\ nm}\tilde{A}_{nm},\end{aligned}\tag{4.80}$$

$$\begin{aligned}&\partial_t B^i_k - \beta^l\partial_l B^i_k - f\partial_k b^i + \alpha^2\mu\frac{\tilde{\gamma}^{ij}}{\phi}\left(\partial_k P_j - \partial_j P_k\right) - \alpha^2\mu\tilde{\gamma}^{ij}\tilde{\gamma}^{nl}\left(\partial_k D_{ljn} - \partial_l D_{kjn}\right)\\ &\quad = B^l_k B^i_l,\end{aligned}\tag{4.81}$$

$$\begin{aligned}&\partial_t D_{kij} - \beta^l\partial_l D_{kij} + s\left(-\frac{1}{2}\tilde{\gamma}_{mi}\partial_{(k}B^m_{j)} - \frac{1}{2}\tilde{\gamma}_{mj}\partial_{(k}B^m_{i)} + \frac{1}{3}\tilde{\gamma}_{ij}\partial_{(k}B^m_{m)}\right) + \alpha\partial_k\tilde{A}_{ij}\\ &\quad = \alpha\frac{1}{3}\tilde{\gamma}_{ij}\tilde{\gamma}^{nm}\partial_k\tilde{A}_{nm} + B^l_k D_{lij} + B^l_j D_{kli} + B^l_i D_{klj} - \frac{2}{3}B^l_l D_{kij}\\ &\quad - \alpha\frac{2}{3}\tilde{\gamma}_{ij}D_k^{nm}\tilde{A}_{nm} - A_k\left(\tilde{A}_{ij} - \frac{1}{3}\tilde{\gamma}_{ij}\,\mathrm{tr}\,\tilde{A}\right),\end{aligned}\tag{4.82}$$

$$\begin{aligned}&\partial_t P_k - \beta^l\partial_l P_k - \frac{1}{3}\phi\left(\alpha\partial_k K - \partial_{(k}B^l_{l)}\right) - \frac{1}{3}\alpha\phi\tilde{\gamma}^{nm}\partial_k\tilde{A}_{nm}\\ &\quad = B^l_k P_l + \frac{1}{3}\left(\alpha K - B^l_l\right)P_k + \frac{1}{3}\phi K A_k - \frac{2}{3}\alpha\phi D_k^{\ nm}\tilde{A}_{nm}.\end{aligned}\tag{4.83}$$

Most of the symbols already appear in the context above and are self-explaining. $\mathrm{tr}\,\tilde{A} = \gamma^{ij}\tilde{A}_{ij}$ is the trace of the conformal traceless extrinsic curvature. There are several red terms added to the evolution systems, using the constraints (4.70) above for a more symmetric characteristic matrix. The $g(\alpha)$ function is the one defined in the slicing condition in section 4.1.4, and the "1+log" slicing is recovered when $g(\alpha) = 2/\alpha$. Several new parameters are also introduced for numerical optimisation:

- τ represents the relaxation time to enforce CCZ4 constraints ($\det\tilde{\gamma}_{ij} = 1$ and $\mathrm{tr}\,\tilde{A}_{ij} = 0$);
- e is the cleaning speed for Hamiltonian constraint, following the idea of Dedner et al., 2002;
- $\mu > 0$ in equation (4.81) determines the effect of the terms from the constraints;
- c controls the contribution from some algebraic source terms in Z4 systems. Its default value is 1 when the original CCZ4 is used (Alic et al., 2012).

The standard hyperbolic Gamma driver shift gauge is given above but for some physical scenarios, we also provide the option of the static(zero) shift condition.

The evolution equations above are not completed as we abbreviate some key quantities for readability. The whole system is closed with the evolving quantities using the following list of relations:

$$\partial_k\tilde{\gamma}^{ij} = -2\tilde{\gamma}^{in}\tilde{\gamma}^{mj}D_{knm} := -2D_k^{ij}, \tag{4.84}$$

$$\tilde{\Gamma}^k_{ij} = \tilde{\gamma}^{kl}\left(D_{ijl} + D_{jil} - D_{lij}\right), \tag{4.85}$$

$$\partial_k\tilde{\Gamma}^m_{ij} = -2D_k^{ml}\left(D_{ijl} + D_{jil} - D_{lij}\right) + \tilde{\gamma}^{ml}\left(\partial_{(k}D_{i)jl} + \partial_{(k}D_{j)il} - \partial_{(k}D_{l)ij}\right), \tag{4.86}$$

$$\Gamma^k_{ij} = \tilde{\Gamma}^k_{ij} - \frac{1}{\phi}\tilde{\gamma}^{kl}\left(\tilde{\gamma}_{jl}P_i + \tilde{\gamma}_{il}P_j - \tilde{\gamma}_{ij}P_l\right), \tag{4.87}$$

$$\begin{aligned}\partial_k\Gamma^m_{ij} =& -2D_k^{ml}\left(D_{ijl} + D_{jil} - D_{lij}\right) + \tilde{\gamma}^{ml}\left[\partial_{(k}D_{i)jl} + \partial_{(k}D_{j)il} - \partial_{(k}D_{l)ij}\right] \\ &+ \frac{2}{\phi}D_k^{ml}\left(\tilde{\gamma}_{jl}P_i + \tilde{\gamma}_{il}P_j - \tilde{\gamma}_{ij}P_l\right) - \frac{2}{\phi}\tilde{\gamma}^{ml}\left(D_{kjl}P_i + D_{kil}P_j - D_{kij}P_l\right) \\ &- \frac{1}{\phi}\tilde{\gamma}^{ml}\left[\tilde{\gamma}_{jl}\partial_{(k}P_{i)} + \tilde{\gamma}_{il}\partial_{(k}P_{j)} - \tilde{\gamma}_{ij}\partial_{(k}P_{l)}\right] + \frac{1}{\phi^2}\tilde{\gamma}^{ml}\left(\tilde{\gamma}_{jl}P_iP_k + \tilde{\gamma}_{il}P_jP_k - \tilde{\gamma}_{ij}P_kP_l\right),\end{aligned} \tag{4.88}$$

$$R^m_{ikj} = \partial_k \Gamma^m_{ij} - \partial_j \Gamma^m_{ik} + \Gamma^l_{ij}\Gamma^m_{lk} - \Gamma^l_{ik}\Gamma^m_{lj}, \quad R_{ij} = R^m_{imj}, \tag{4.89}$$

$$\nabla_i \nabla_j \alpha = \partial_{(i}A_{j)} - \Gamma^k_{ij}A_k, \tag{4.90}$$

$$\partial_k \tilde{\Gamma}^i = -2D_k^{jl}\tilde{\Gamma}^i_{jl} + \tilde{\gamma}^{jl}\partial_k\tilde{\Gamma}^i_{jl}, \tag{4.91}$$

$$Z_i = \frac{1}{2}\tilde{\gamma}_{ij}\left(\hat{\Gamma}^j - \tilde{\Gamma}^j\right), \quad Z^i = \frac{1}{2}\phi^2\left(\hat{\Gamma}^i - \tilde{\Gamma}^i\right), \tag{4.92}$$

$$\nabla_i Z_j = D_{ijl}\left(\hat{\Gamma}^l - \tilde{\Gamma}^l\right) + \frac{1}{2}\tilde{\gamma}_{jl}\left(\partial_i\hat{\Gamma}^l - \partial_i\tilde{\Gamma}^l\right) - \Gamma^l_{ij}Z_l, \tag{4.93}$$

$$R + 2\nabla_k Z^k = \phi^2\tilde{\gamma}^{ij}\left(R_{ij} + \nabla_i Z_j + \nabla_j Z_i\right). \tag{4.94}$$

Notice the second-order constraints (4.69) are used to “symmetrise” the spatial derivatives of the auxiliary variables in some equations and relations. In code practice, we also use a simplified expression for the modified Ricci tensor $R_{ij} + 2\nabla_{(i}Z_{j)}$ from Radia et al., 2022:

$$\begin{aligned} R_{ij}+2\nabla_{(i}Z_{j)} = \\ & -\frac{1}{2}\tilde{\gamma}^{kl}\partial_k\partial_l\tilde{\gamma}_{ij} + \tilde{\gamma}_{k(i}\partial_{j)}\hat{\Gamma}^k + \frac{1}{2}\hat{\Gamma}^k\partial_k\tilde{\gamma}_{ij} + \tilde{\gamma}^{lm}\left(\tilde{\Gamma}^k_{li}\tilde{\Gamma}_{jkm} + \tilde{\Gamma}^k_{lj}\tilde{\Gamma}_{ikm} + \tilde{\Gamma}^k_{im}\tilde{\Gamma}_{klj}\right) \\ & + \frac{1}{\phi}\left(\tilde{\nabla}_i\tilde{\nabla}_j\phi + \tilde{\gamma}_{ij}\tilde{\gamma}^{kl}\tilde{\nabla}_k\tilde{\nabla}_l\phi\right) - \frac{2}{\phi^2}\tilde{\gamma}_{ij}\tilde{\gamma}^{kl}\partial_k\phi\partial_l\phi \\ & + \frac{2}{\phi^3}Z^k\left(\tilde{\gamma}_{ik}\partial_j\phi + \tilde{\gamma}_{jk}\partial_i\phi - \tilde{\gamma}_{ij}\partial_k\phi\right), \end{aligned} \tag{4.95}$$

with $\tilde{\nabla}_i\tilde{\nabla}_k\phi = \partial_i\partial_k\phi - \tilde{\Gamma}^k_{ij}\partial_k\phi$ the second *conformal* covariant derivative of ϕ and $\tilde{\Gamma}_{kij} := \tilde{\gamma}_{kl}\tilde{\Gamma}^l_{ij}$ the newly-defined lowered Christoffel symbol.

The final evolution system of FOCCZ4 contains 58 independent variables as

$$\left\{\tilde{\gamma}_{ij}, \alpha, \beta^i, \phi, \tilde{A}_{ij}, K, \Theta, \hat{\Gamma}^i, b^i, A_k, B^i_k, D_{kij}, P_k\right\}. \tag{4.96}$$

The complete list of evolving components is given in Appendix C. An example of the characteristic matrix of the system is given in figure 4.3. The terms that appear in the pure recast of CCZ4 are plotted as blue dots while the extra terms added utilising constraints (4.69) and (4.70) are plotted as red dots. One can see clearly how the improved FOCCZ4 formulation helps enhance the symmetry of the matrices. The approximately symmetric characteristic matrix leads to strong hyperbolicity of

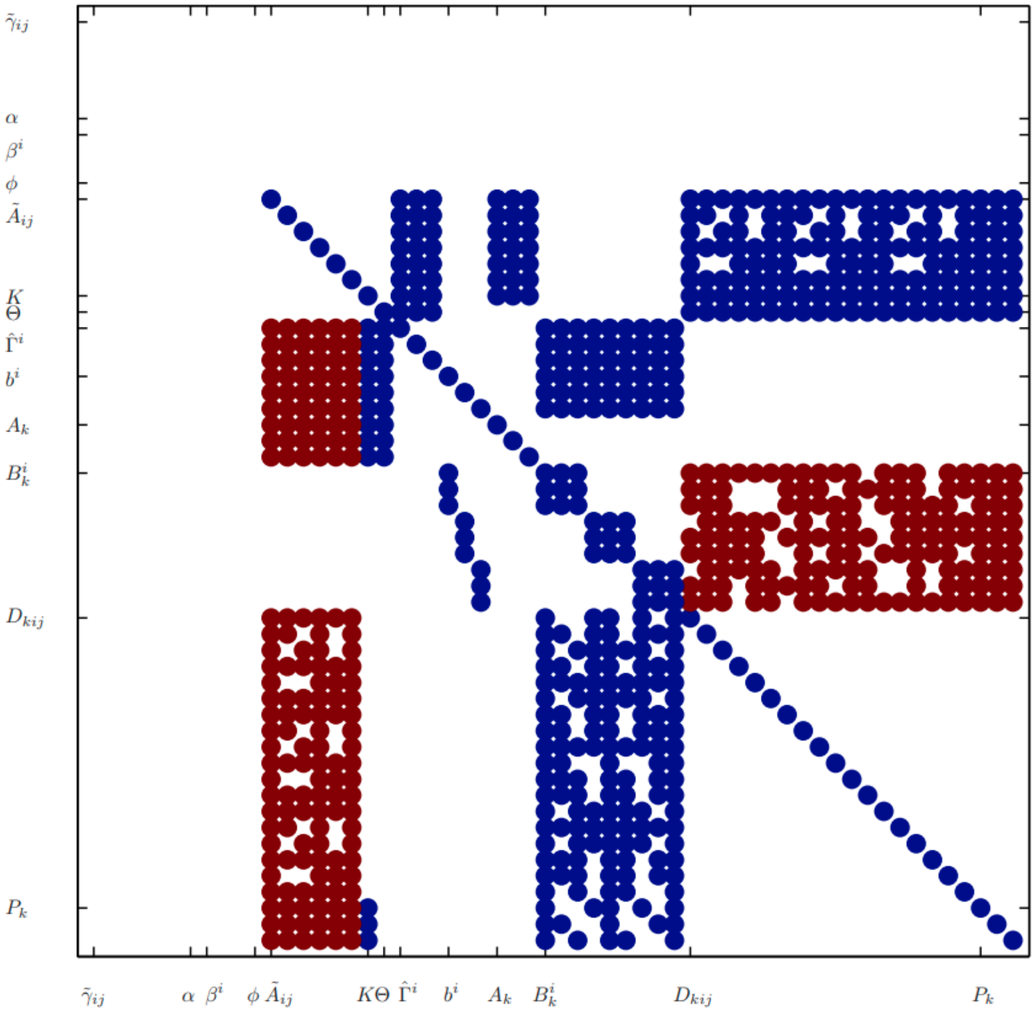

Figure 4.3: The sparsity pattern of the characteristic matrix of FOCCZ4 system, with the terms from the pure recast of CCZ4 plotted as blue dots and the extra terms added utilising constraints (4.69) and (4.70) plotted as red dots. The Gamma driver and 1+log gauge conditions are used and the normal vector for the characteristic matrix is $\mathbf{n} = (1, 1, 1)/\sqrt{3}$. The pattern shows an approximate symmetry with the help of the extra terms, hence giving the evolving system a strong hyperbolicity. The figure is cited from Dumbser et al., 2018.

the FOCCZ4 system as all its eigenvalues are real and the eigenvectors are linearly independent. We also compute the eigenvalues of the matrix for wave speed which is used in some numerical kernels. For a more thorough analysis of this hyperbolicity and eigen-property of the FOCCZ4 system, see Dumbser et al., 2018.

4.1.4 Gauge condition

As we mentioned in Section 4.1.1, the gauge condition describes how our hypersurfaces (as well as the coordinate system on it) alter through spacetime. We can control this by specifying evolution equations of the gauge quantities lapse α and shift β^i. These gauge quantities carry four degrees of coordinate freedom, i.e., the choice of different gauge systems has no impact on the actual physical process; rather, it simply represents how the spacetime is decomposed.

Finding a proper gauge condition for various astrophysics processes is not trivial and requires physical insights and numerical experiments. One significant property of a well-behaved gauge condition is the ability to prevent (both coordinate and physical) singularities from arising. Singularities in numerical simulations can be disastrous and will lead to the infinity of one or more variables and eventually crash of code.

With a proper treatment in the initial condition (see section 4.1.2), the pre-existing singularities in the simulation can be avoided. However, if we do not handle the gauge condition correctly, new singularities may appear during the evolution due to the attractive nature of gravitational sources. One may see this point more clearly by considering the simplest and most straightforward gauge condition:

$$\alpha = 1, \quad \beta^i = 0. \tag{4.97}$$

This is called *geodesic gauge*. In this gauge, all coordinates are static for observers travelling along the normal direction. because there is no shift, and their coordinates time coincides with the proper time as the lapse is unity. Therefore, the coordinate lines actually follow free-falling geodesics, which is where the name of this gauge condition comes from. This gauge soon forms a singularity as long as there are gravitational sources presented. It is not surprising intuitively, as geodesics, thus the coordinate lines, tend to focus through positively curved spacetime. Equations (4.28) and (4.29) become

$$\partial_t K = K_{ij}K^{ij} + 4\pi(\rho + 3P) \geq 0, \tag{4.98}$$

$$\partial_t \ln \gamma^{1/2} \quad = \quad -K, \tag{4.99}$$

under this gauge. The first equation shows that the trace of the extrinsic curvature is self-enhanced under the geodesic gauge, and the second equation tells us that the coordinate volume elements approach zero given a positive trace of extrinsic curvature. It is highly unlikely to achieve a stable evolution in any meaningful scenario with this gauge condition.

Various gauge conditions have been discovered for different astrophysical processes. Here we only describe the most widely-used one in black hole simulations: the *1 + log* time slicing for α and the *Gamma driver* spatial gauge for β^i. We adopt these gauge conditions in our black hole simulation application.

The evolution equation of α gains its name as a "time-slicing condition" because this field describes the time lapse speed through the hypersurfaces. Lower values of α indicate a slower passage of proper time, while higher values indicate a faster passage of proper time. A naive choice one may consider to avoid singularities is to make the proper time in high-curvature regions tick slower, i.e., split more hypersurfaces as the observer gets closer to the singularity. It indicates a gauge condition that takes the form of $\mathcal{L}_{\mathbf{t}}\alpha = -\mathcal{F}K$, where $\mathcal{F}$ is a coefficient that may depend on other quantities. This kind of gauge condition can be derived by defining the so-called *harmonic coordinates* (Smarr and York, 1978):

$${}^{(4)}\Gamma^a := g^{bc}\ {}^{(4)}\Gamma^a_{bc} = -\frac{1}{|g|^{1/2}}\partial_b(|g|^{1/2}g^{ab}) = 0. \tag{4.100}$$

This is a vector equation, so it fixes the four gauge freedoms completely. Usually, we only apply the time components of this equation, which is also named as *harmonic slicing* in this case. Putting the metric formulation (4.21) back to the equation above, we get our explicit expression of time slicing:

$$(\partial_t - \beta^i \partial_i)\alpha = -\alpha^2 K, \tag{4.101}$$

which is in the form $\mathcal{L}_{\mathbf{t}}\alpha = -\mathcal{F}K$ that we have guessed above. This gauge condition gives a much more stable evolution in many numerical experiments while remaining in a rather simple formulation compared to the geodesic condition (e.g., Shibata and Nakamura, 1995b). To further generalise the condition, we can add an extra factor on the right of the expression above (Bona et al., 1995):

$$(\partial_t - \beta^i \partial_i)\alpha = -\alpha^2 f(\alpha) K, \tag{4.102}$$

where $f(\alpha)$ is a positive scalar function and only depends on the lapse. One important limiting form is $f(\alpha) \to \infty$ when this harmonic slicing gauge condition approximates the *maximal slicing* condition. The maximal slicing condition asks for a vanished trace of extrinsic curvature all the time, which gives better behaviour with the price of solving elliptic equations during evolution. The infinite factor f guarantees any region on the hypersurface with a non-zero K does not evolve along the proper time.

Clearly, we cannot apply an evolution equation that contains infinite factors in our numerical experiments. Instead, we can try a formulation that approximates the behaviour. A good choice is to set $f(\alpha) = 2/\alpha$, which gives

$$(\partial_t - \beta^i \partial_i)\alpha = -2\alpha K. \tag{4.103}$$

Using equation (4.29) and assuming a zero shift, one can integrate to get

$$\alpha = 1 + \ln \gamma. \tag{4.104}$$

We have chosen the boundary condition as $\alpha|_{\gamma=1} = 1$ above. This is why this gauge condition is widely called 1+log slicing. This gauge condition has proven to have a quite good property in singularity avoidance, and it has been very successful in simulating black hole spacetime with the puncture method that we mentioned in section 4.1.2. Its good property can be seen intuitively from the observation that a small spatial volume element gives a small α and further results in a larger $f(\alpha)$. So, it behaves in a positive feedback style and is more like maximal slicing than harmonic slicing in the region with high extrinsic curvature (Baumgarte and Shapiro, 2010).

The shift gauge that we are going to apply is based on an analogous definition of harmonic coordinates but in three dimensions and using the conformal Christoffel symbol defined in section 4.1.2:

$$\bar{\Gamma}^i := \bar{\gamma}^{jk}\bar{\Gamma}^i_{jk} = -\partial_j\bar{\gamma}^{jk}. \tag{4.105}$$

The second equality uses the fact that $\bar{\gamma} = \det(\bar{\gamma}_{ij}) = 1$. One can require the time derivative of this connection function $\bar{\Gamma}^i$ to vanish to get the *Gamma freezing* spatial gauge, which finally results in an elliptic equation similar to the one from maximal slicing (Alcubierre and Brügmann, 2001). The Gamma freezing condition is quite well-behaved but expensive numerically; therefore, many applications choose to approximate its behaviour with the so-called *Gamma driver* gauge condition (Alcubierre and Brügmann, 2001; Duez et al., 2003):

$$\partial_t\beta^i = f(\partial_t\bar{\Gamma}^i + \eta\bar{\Gamma}^i). \tag{4.106}$$

To enhance the hyperbolicity, one can further construct the spatial gauge condition as (Alcubierre et al., 2003a)

$$\begin{aligned} (\partial_t - \beta^j\partial_j)\beta^i &= f b^i, && (4.107)\\ (\partial_t - \beta^j\partial_j)b^i &= (\partial_t - \beta^j\partial_j)\bar{\Gamma}^i - \eta b^i. && (4.108) \end{aligned}$$

This set of equations of β^i and new support field b^i are called *hyperbolic Gamma-driver* to distinguish it from the original parabolic one. f and η are parameters that may vary from problem to problem, and their choices are somewhat empirical. Our simulations typically use $f = 0.75$ and $\eta = 1$, which is consistent with previous studies of similar evolution systems in the literature (Alic et al., 2012; Dumbser et al., 2018).

The hyperbolic gamma driver spatial gauge (4.108), combined with the 1+log time slicing (4.103), are widely used in evolving puncture black hole spacetimes, and they show the ability to give long-term stable and well-behaved evolution (Campanelli et al., 2006). We adopt these gauge conditions in the following content of this chapter unless specifically stated otherwise.

4.1.5 Gravitational wave extraction

The sections above cover aspects of simulating black hole spacetimes as a Cauchy initial value problem. Therefore we now have, in principle, all the theoretical tools we need to build our application. However, one may already notice that we make many assumptions and choices mainly based on numerical considerations: we utilise our freedom to choose the structure of the spacetime foliation, the shape of the initial hypersurface, and the coordinate allocation on the spacetime slices, etc... Most of them are there for stability enhancement and singularity avoidance. It is reasonable to do so, as different choices of slicing and coordinates just affect how we cut through spacetime, the actual physical process remains unaffected. As long as the interested domain in spacetime is fully covered by the simulation, we can adjust how hypersurfaces are integrated as needed.

Though this freedom is quite useful from a numerical perspective, it requires us to be more careful in taking full advantage of the simulations. As we mentioned in the Introduction, any meaningful astrophysics simulations should be able to reproduce observations and make predictions for the targeting physical process, and this is usually achieved by outputting quantities that are also available in the real observations. These observable quantities should be independent of coordinates and calculated in the frame of an infinite observer (Earth). Different observable quantities can be selected for various astrophysics scenarios. In the case of black holes, particularly rotating binary black holes, the gravitational wave is the most important one.

There are several ways to extract gravitational waves from a black hole spacetime, and here we focus on the widely-used *Newman-Penrose approach* (Newman and Penrose, 2004) that we will follow in our code. The extraction of this method is achieved by calculating the so-called Newman-Penrose scalars $\psi_0, \psi_1, \ldots, \psi_4$ on a distant sphere from the source (black holes). Those quantities are naturally coordinate-independent as they are scalar fields. The Newman-Penrose scalars represent the 10 independent components of the (4-dimensional) Weyl tensor and are from the contraction of the Weyl tensor and a null tetrad (four null vectors that form a complete basis). We can properly choose our null tetrad such that the scalars ψ_1 and ψ_3

vanish, while ψ_0 and ψ_4 measure the ingoing and outgoing gravitational radiation. ψ_2, on the other hand, gives the longitudinal part of the gravitational waves. We are mainly working on the ψ_4 scalar in our content for obvious reasons.

The choice of a proper null tetrad is not unique, and the notation for those vectors also varies in the literature. Here we follow the formulation in Baumgarte and Shapiro, 2010 in which it is defined as defined as:

$$\begin{aligned} l^a &= \frac{1}{\sqrt{2}}\left(e^a_{\hat{t}} + e^a_{\hat{r}}\right), && (4.109)\\ k^a &= \frac{1}{\sqrt{2}}\left(e^a_{\hat{t}} - e^a_{\hat{r}}\right), && (4.110)\\ m^a &= \frac{1}{\sqrt{2}}\left(e^a_{\hat{\theta}} + ie^a_{\hat{\phi}}\right), && (4.111)\\ \bar{m}^a &= \frac{1}{\sqrt{2}}\left(e^a_{\hat{\theta}} - ie^a_{\hat{\phi}}\right), && (4.112) \end{aligned}$$

where $e^a_{\hat{t}}, e^a_{\hat{r}}, e^a_{\hat{\theta}}, e^a_{\hat{\phi}}$ are unit vectors in each direction under the 4 dimensional metric g_{ab}. The tetrad is constructed such that the only non-vanishing inner products between them are:

$$-l^a k_a = m^a \bar{m}_a = 1. \tag{4.113}$$

Now we use this tetrad to contract the Weyl vector for ψ_4 (the sign convention is also not universal, different signs are used in the literature):

$$\psi_4 =^{(4)} C_{abcd} k^a \bar{m}^b k^c \bar{m}^d. \tag{4.114}$$

The upper index $^{(4)}$ indicates it is a 4-dimensional quantity. The Weyl tensor is defined as

$$\begin{aligned} {}^{(4)}C_{abcd} =& {}^{(4)}R_{abcd} - \frac{1}{2}\left(g_{ac}{}^{(4)}R_{bd} - g_{ad}{}^{(4)}R_{bc} - g_{bc}{}^{(4)}R_{ad} + g_{bd}{}^{(4)}R_{ac}\right) \\ &+ \frac{1}{6}\left(g_{ac}g_{bd} - g_{ad}g_{bc}\right){}^{(4)}R. \end{aligned} \tag{4.115}$$

Combining eq. (4.109)-(4.112), (4.114) and (4.115) we get

$$\begin{aligned}\psi_4 = -\frac{1}{4}\Big({}^{(4)}R_{\hat{t}\hat{\theta}\hat{t}\hat{\theta}} - 2i^{(4)}R_{\hat{t}\hat{\theta}\hat{t}\hat{\phi}} - 2^{(4)}R_{\hat{t}\hat{\theta}\hat{r}\hat{\theta}} + 2i^{(4)}R_{\hat{t}\hat{\phi}\hat{r}\hat{\theta}} - {}^{(4)}R_{\hat{t}\hat{\phi}\hat{t}\hat{\phi}} \\ + {}^{(4)}R_{\hat{r}\hat{\theta}\hat{r}\hat{\theta}} + 2i^{(4)}R_{\hat{t}\hat{\theta}\hat{r}\hat{\phi}} + 2^{(4)}R_{\hat{t}\hat{\phi}\hat{r}\hat{\phi}} - 2i^{(4)}R_{\hat{r}\hat{\phi}\hat{r}\hat{\theta}} - {}^{(4)}R_{\hat{r}\hat{\phi}\hat{r}\hat{\phi}}\Big).\end{aligned} \tag{4.116}$$

Now consider the large radius approximation. When we are far away from the source, the perturbation $g_{ab} \approx \eta_{ab} + h_{ab}$, $h_{ab} \ll 1$ can be applied. It gives a linearized Riemann tensor:

$${}^{(4)}R_{abcd} = \frac{1}{2}\left(\partial_a\partial_d h_{bc} + \partial_b\partial_c h_{ad} - \partial_b\partial_d h_{ac} - \partial_a\partial_c h_{bd}\right). \tag{4.117}$$

We further assume the *transverse-traceless* gauge in a spherical coordinate (Auger and Plagnol, 2017) and a static observer, i.e., its 4-velocity does not have formspatial component, $\gamma_{ab}u^b = 0$. In this frame, the perturbed part (wave part) of an outgoing gravitational wave can be written as:

$$h_{ab}^{\mathrm{TT}} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & h_+ & h_\times \\ 0 & 0 & h_\times & -h_+ \end{pmatrix}. \tag{4.118}$$

The only non-zero components of h_{ab} are $h_{\hat{\theta}\hat{\theta}} = -h_{\hat{\phi}\hat{\phi}}$ and $h_{\hat{\theta}\hat{\phi}} = h_{\hat{\phi}\hat{\theta}}$. We also know an outgoing wave at a large distance satisfies $h_{ab}(t, r, \theta, \phi) = h_{ab}(t - r, \theta, \phi)$, thus $\partial_r h_{ab} = -\partial_t h_{ab}$. Using these relations and eq. (4.116), the Newman-Penrose scalar ψ_4 can be expressed as

$$\psi_4 = -\partial_t^2 h_+ + \partial_t^2 h_\times. \tag{4.119}$$

Here we can see the real and the imaginary parts of ψ_4 are measurements of the two modes of an outgoing gravitational wave respectively. We can further decompose ψ_4 into a superposition of modes with the base $s = -2$ spin-weighted spherical harmonics ${}_{-2}Y_{lm}$:

$$\psi_4(t, r, \theta, \phi) = \sum_{l=2}^{\infty} \sum_{m=-l}^{l} \psi_4^{lm}(t, r)\,{}_{-2}Y_{lm}(\theta, \phi), \tag{4.120}$$

where the coefficients (the mode strength) are found by the inner product

$$\psi_4^{lm} = \int_0^{2\pi}\int_0^{\pi} {}_{-2}Y_{lm}^{*}\psi_4 \sin\theta d\theta d\phi. \tag{4.121}$$

For a binary black hole merger, the dominating mode is $l = 2, m = 2$, and its corresponding base function is

$${}_{-2}Y_{22} = \sqrt{\frac{5}{64\pi}}(1+\cos\theta)^2 e^{2i\phi}. \tag{4.122}$$

In code practice, we can calculate ψ_4 directly from its definition (4.116) (e.g., Baker, Campanelli, and Lousto, 2001), nevertheless, a further simplification can help reduce the computational burden. We first introduce the so-called electric and magnetic parts of the Weyl tensor:

$$E_{ab} := n^c n^d \, {}^{(4)}C_{cadb}, \tag{4.123}$$

$$B_{ab} := n^c n^d \, {}^{(4)}C^{*}_{cadb}, \tag{4.124}$$

where C^{*}_{cadb} is the dual Weyl tensor and defined as

$${}^{(4)}C^{*}_{cadb} := \frac{1}{2}\epsilon^{ef}{}_{db}C_{caef}, \tag{4.125}$$

where ϵ_{abcd} is the four-dimensional volume form. It can be proved that the electric and magnetic parts are both symmetric, traceless and spacelike, so they each carry 5 independent components of the original Weyl tensor. With these two new tensors, the scalar ψ_4 reduces to (Alcubierre, 2008):

$$\psi_4 = (E_{ij} - iB_{ij})\bar{m}^i\bar{m}^j. \tag{4.126}$$

The sum in the expression above is 3-dimensional because all quantities are spatial, thus all information is included in their spatial part again. Using Gauss–Codazzi (4.15) and Codazzi–Mainardi (4.16) equations, the electric and magnetic parts of the Weyl tensor read as

$$E_{ij} = \left(R_{ij} - K^m_{\ i} K_{jm} + K_{ij}(K - \Theta) + D_{(i}Z_{j)} - 4\pi S_{ij}\right)^{TF}, \tag{4.127}$$

$$B_{ij} = \frac{1}{2}\left(\epsilon_{mni} D^m K_j^{\ n} + \epsilon_{mnj} D^m K_i^{\ n}\right)^{TF}, \tag{4.128}$$

where TF is the trace-free index which has the same definition as in section 4.1.3. As we are considering the vacuum solution, the source term in equation (4.127) is zero.

We then construct the null tetrad in our foliation of hypersurfaces. The unit vectors in three spatial directions are derived from a Gram-Schmidt orthonormalization of the following vectors (Brügmann et al., 2008):

$$v^i = [-y, x, 0], \tag{4.129}$$

$$u^i = [x, y, z], \tag{4.130}$$

$$w^i = \gamma^{ij}\epsilon_{jkl} v^k u^l. \tag{4.131}$$

which become

$$v^i \rightarrow v^i/\sqrt{L_{vv}}, \tag{4.132}$$

$$u^i \rightarrow (u^i - v^i L_{vu})/\sqrt{L_{uu}}, \tag{4.133}$$

$$w^i \rightarrow (w^i - v^i L_{vw} - u^i L_{uw})/\sqrt{L_{ww}}, \tag{4.134}$$

where L represents the inner product of corresponding vectors, i.e., $L_{vu} := \gamma_{ij} v^i u^j$. The unit vector for the time dimension can be written as $e^a_{\hat{t}} = [\alpha^{-1}, -\alpha^{-1}\beta^i]$ in the standard 3+1 language, thus, the tetrad we use are:

$$l^a = \frac{1}{\sqrt{2}}[\alpha^{-1}, -\alpha^{-1}\beta^i + u^i], \tag{4.135}$$

$$k^a = \frac{1}{\sqrt{2}}[\alpha^{-1}, -\alpha^{-1}\beta^i - u^i], \tag{4.136}$$

$$m^a = \frac{1}{\sqrt{2}}[0, w^i + iv^i], \tag{4.137}$$

$$\bar{m}^a = \frac{1}{\sqrt{2}}[0, w^i - iv^i]. \tag{4.138}$$

We only need $\bar{m}^i$ when calculating ψ_4. Notice all ϵ that appear above are volume form, i.e., the Levi-Civita tensors and related to the normal Levi-Civita symbols ε_{ijk} with:

$$\epsilon_{ijk} = \sqrt{|\det \gamma_{ij}|}\varepsilon_{ijk}, \quad \epsilon_{ijk} = \frac{\operatorname{sgn}(\det \gamma_{ij})}{\sqrt{|\det \gamma_{ij}|}}\varepsilon_{ijk} = \frac{1}{\sqrt{|\det \gamma_{ij}|}}\varepsilon_{ijk}. \tag{4.139}$$

In the wave extraction zone, the spatial metric is close to the flat one, thus its determinant should be positive. In our application, we calculate the E_{ij}, B_{ij} and $\bar{m}^i$ from equations (4.127), (4.128) and (4.138), then put them back in equation (4.126) to get the real and imaginary parts of the ψ_4.

The last step of the extraction is the spherical integration. The numerical methods for this are widely studied in the literature (e.g., Trefethen, 2012; Press et al., 2007). As our particle tracers can be placed at arbitrary positions in the simulation domain, we adopt the spherical t-design scheme (Brauchart and Grabner, 2015) for our integration. This method requires the values of the integrated function on a set of N sample points on the targeted sphere. The coordinates of the sample points are pre-calculated and can be looked up in the tables. It is shown that the average of those sample values is also the average of the integrated function itself, i.e.,

$$\int_0^{2\pi}\int_0^{\pi} f(\mathbf{x})\sin\theta d\theta d\phi = \frac{4\pi}{N}\sum_0^{N-1} f(\mathbf{x}_i), \quad \forall f(\mathbf{x}_i) \in \Pi^t. \tag{4.140}$$

Here Π^t represents the space spanned by polynomials to the degree of t. This method works best when the function of interest has a degree of freedom under t. N needs to be increased if we want higher accuracy by raising t. In our code, we choose the set of sample points of $t = 43$ with $N = 948$ which is given in the datasets from Womersley, 2015.

4.2 Code Implementation

In this section, we will present a description of the code implementation of our numerical relativity application. We will not present the technical details of building applications here as they have been covered in chapter 2. This section focuses on the code segments and modules we use in the application and how we combine them together in the code practices. When applicable, we will refer to the relevant sections in chapter 2 for the mentioned modules in the following content.

A diagram showing the structure of our application is given in figure 4.4, arranged according to the workflow of a complete simulation. The white blocks on the left indicate the different stages of the code execution and the texts left to them describe processes occurring in each phase. The purple blocks on the right of the diagram show the different code modules involved in the respective stages of the simulation. We will go through them in the content below and introduce them one by one in the same order as they appear in the diagram.

4.2.1 Initial condition Stage

This stage is where we choose our physical process in the simulation. All scenarios of our application are vacuum solutions currently, so we only need to specify the initial spacetime structure, including the initial hypersurface layout and the gauge conditions. The running parameters are also sent to the code when the corresponding scenario is picked. We will discuss the parameter choices in the result section 4.3.

- **Direct Assignment**. The rather simple physical scenarios, including the (diagonal) gauge wave and the linear wave, can be specified by directly assigning the initial values of the evolving variables. A detailed description of those setups is given in Appendix C. For the gauge condition, the harmonic slicing, which set $f(a) = 1$ in equation 4.72, and the static zero shift are adopted in those setups.

- **TwoPuncture Library**. As we introduced in section 4.1.2, the initial condition of puncture black hole spacetime requires a numerical solution of the Hamiltonian constraints. Therefore, a direct approach is not practical. We

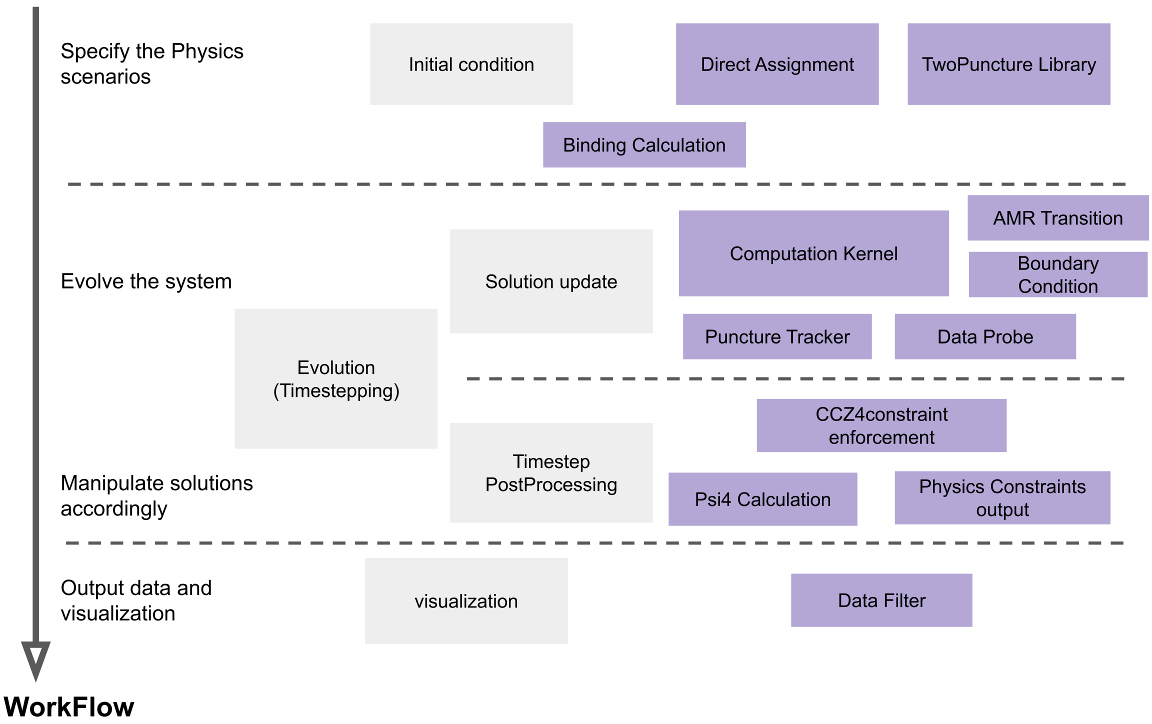

Figure 4.4: The illustration of the code structure organised in the workflow. The white blocks on the left and the texts next to them describe different stages during a simulation, while the purple blocks on the right show the code modules we utilise in each respective stage. We go through those segments following the order here in the main content.

ported the TwoPuncture module from the numerical library EINSTEINTOOLKIT. This external library receives physical parameters of the black hole systems like the mass, linear and angular momentum of the black holes, the distance between black holes, etc... It then solves the Hamiltonian constraints numerically and provides the spacetime quantities γ_{ij}, K_{ij} and α that we need. We still need to specify our gauge conditions while using this library, and we choose the widely-used combination of 1+log slicing and Gamma driver shift gauge in our black hole simulations.

- **Binding Calculation**. This code segment is only used in simulations of black hole spacetimes, where we receive initial conditions from the TwoPuncture library. This external module only gives the basic ADM quantities, and some extra calculation is needed to convert them into the evolving quantities in our FOCCZ4 system. We first calculate the conformal metric $\tilde{\gamma}_{ij}$, two parts of the extrinsic curvature $\tilde{A}_{ij}$ and K, also the conformal metric ϕ. Then we use the fourth-order finite difference to compute the first derivative of those quantities, which are used to assign values to the auxiliary variables (4.68).

4.2.2 Evolution: Solution update

The actual temporal integral happens at this stage. A chosen computational kernel is applied to the solution every timestep and the treatment of the boundary condition and the resolution transition also happens here. There are also some on-the-fly extension modules that do their work at this stage.

- **Computation Kernel**. The computation kernel refers to the numerical scheme we choose to apply to our evolution system. Different computation kernels handle the PDE system we specify in section 2.1.3 differently, and we tested several numerical kernels for our numerical relativity application. Currently, we are using the FD4-RK1-AdaptiveTimeStep kernel, which refers to the scheme of fourth-order finite difference, first-order Runge-Kutta (Euler) and adaptive timestep size based on the maximal wave speed (eigenvalues). The details of this solver are explained in section 2.2.2. We use the lopsided finite difference stencil for the advection terms, following the idea of Radia et al., 2022. The finite volume solver in section 2.2.1 turns out to be too dissipating and not suitable in black hole simulations, and we also find that a higher Runge-Kutta order does not help much in improving simulation quality in our tests.

- **AMR Transition**. This code segment is responsible for resolution transition, i.e., the interpolation and the restriction of the data at those boundaries. Due to the nature of our FD4 kernel, we need to fill a halo of three layers of ghost volumes, and it is currently achieved by the tensor product scheme described in section 2.4.2 with an accuracy of linear order. We are also developing other methods for AMR transition which provide higher-order accuracy for future simulations.

- **Boundary Condition**. Similar to the AMR transition, we need to fill three halo layers of ghost volumes on the boundary of our simulation domain. For the simple wave scenarios, the straightforward periodic boundary condition is applied. On the other hand, we provide two boundary conditions for the black hole simulations. The first is the homogeneous Newmann condition which eliminates the derivatives of every quantity – this condition is rather simple in implementation but yields a quite unstable boundary. Thus we provide the Sommerfeld-type boundary condition described in section 2.3.2 and it shows improved performance in both single and binary black hole scenarios.

- **Puncture Tracker**. This module is developed to track the black holes during the evolution of binary black holes. During every timestep, we first linearly interpolate the corresponding quantities from the nearest eight volumes to compute the shift vector on each puncture position, then a simple explicit Euler solver is implemented to integrate the shift vector in the other direction

$$\frac{\mathrm{dx^i}}{\mathrm{dt}} = -\beta^i \quad \rightarrow \quad x^i_{t+\delta t} = x^i_t - \delta t \cdot \beta^i_t. \tag{4.141}$$

  The new positions of the punctures are updated accordingly. This extension is not optimised within the framework of ExaHyPE2 and will soon be replaced by the data probe we present below.

- **Data Probe**. The module utilised the particle support of ExaHyPE2. We present this feature in section 2.5. By freezing particle tracers in the domain, we can use them to probe the solution at certain positions, which is very helpful in temporal plotting. We also utilise those static probes to do the spherical integral in 4.1.5, combining with the ψ_4 calculation module below. When we set the initial positions to coincide with the punctures and ask them to follow the negative shift vector, those particle tracers behave like the puncture tracker we described above. As the particle module of ExaHyPE2 is more advanced in code development, we plan to switch to this feature for black hole tracing after further tests in the future.

4.2.3 Evolution: TimeStep PostProcessing

After the solution update, there are still several things that need to be done within the scope of a single timestep. Algebraic constraints are enforced explicitly, aiming to clear the accumulating numerical errors, and extra quantities are calculated in this stage. Those quantities are attached to the evolving solutions and provide more output information from the simulations.

- **CCZ4 constraint enforcement**. This segment is designed to apply the algebraic constraints of the FOCCZ4 system. Those constraints also appear in the original CCZ4 formulation so we do not specify the First Order in its name. The enforced constraints include the tracelessness of the $\tilde{A}_{ij}$, the unity of the $\det(\tilde{\gamma}_{ij})$ and equation (4.70). This enforcement is applied after the solution is updated in a timestep and the corresponding evolving variables are over-written to satisfy the constraints. The traceless property is re-achieved by removing the trace from every component of the tensor, and the unity of determinant is re-achieved by rescaling every component.

- ψ_4 **Calculation**. The most important extra quantities are clearly the real and imaginary parts of ψ_4 that represent the gravitational wave signal. We use second-order finite difference to compute the derivatives of evolving variables and then calculate ψ_4 following the methods in section 4.1.5. The temporal behaviour of this quantity at specific locations in the domain can be extracted by the data probes above, while its mode components from the spherical integral can also be calculated using corresponding probes.

- **Physics Constraints Output**. This module is implemented similarly to the ψ_4 Calculation, but we output the Hamiltonian and momentum constraints instead this time. They are important physical quantities used to monitor the correctness and accuracy of a numerical relativity simulation. The calculation of those extra quantities is separated from the computational kernel and thus has a rather big freedom of variables outputted. We can examine the intermediate quantities, such as the Ricci tensor R_{ij} or conformal connection function $\tilde{\Gamma}^i$, to help us debug during the code tests.

4.2.4 Visualization

The visualization of the output is the last step of the workflow. Fewer notable processes occur during this stage, but it plays an essential role in presenting the results visually for analysis. The raw data outputted by ExaHyPE2 is in its specific patch format, and every rank plots its output file to reduce the communication cost. After a straightforward merge-convert process, the output can be transformed into the standard VTU format, which is compatible with various visualization tools. The data from the particle probe, on the other hand, is provided in CSV form, allowing for processing via standard data visualization. In the result section 4.3 below we will present test results based on these two types of outputs.

- **Data Filter**. This code segment is designed to provide more control on the output snapshots. Given the scale of the simulations we conduct and the number of evolving variables, the resulting output files can be quite large, thus causing difficulty in loading and manipulating in the visualization tools. Therefore we incorporated a data filter into our code. This filter enables us to selectively extract slices and clips from the domain, allowing us to focus only on the outputs of interest. This feature is very helpful especially when our simulating phenomenon has some intrinsic symmetry, e.g., the single black hole scenario. In principle, we only need to plot one-eighth of the whole domain for a simulation of the black hole at the origin. The left seven eighth of the domain can be derived according to the spherical symmetry. In code practice, we usually output the slices over the x-y plane where the single black hole sits and the binary black holes rotate and this approach significantly reduces the size of the output files by a factor of ten. We can also specify what variables are included in our output snapshot. As more than half of the evolving quantities are auxiliary variables and have some symmetries, there is no need to visualise all of them. We pick different outputted variables during the code development for analysis, and the most important quantities we always monitor closely include the conformal factor ϕ, the first entry of conformal metric $\tilde{\gamma}_{11}$ and the time component of the Z4 vector, Θ.

4.3 Simulation Tests

In this section, we report the simulation results of our application on various physical scenarios. We begin with a standard gauge wave simulation and then proceed to examine the code performance in the black hole spacetime, covering the single Schwarzschild black hole, the head-on colliding binary black holes and the rotating binary black hole merger.

4.3.1 Gauge Wave

The gauge wave scenario is one of the standard test cases for numerical relativity codes (Alcubierre et al., 2003b), where a flat Minkowski spacetime is considered. No actual physical phenomenon occurs in the system, however, we now slice our spacetime dynamically by performing a time-dependent coordinate transformation:

$$\hat{t} = t - \frac{A}{2k\pi}\cos[k\pi(x-t)], \tag{4.142}$$

$$\hat{x} = x + \frac{A}{2k\pi}\cos[k\pi(x-t)], \tag{4.143}$$

$$\hat{y} = y, \tag{4.144}$$

$$\hat{z} = z. \tag{4.145}$$

which converts the original Minkowski metric $ds^2 = -d\hat{t}^2 + d\hat{x}^2 + d\hat{y}^2 + d\hat{z}^2$ into the new formulation as

$$ds^2 = -H(x,t)dt^2 + H(x,t)dx^2 + dy^2 + dz^2, \quad H(x,t) = 1 - A\sin[k\pi(x-t)]. \tag{4.146}$$

It represents a gauge wave propagating along the x-axis with an amplitude of A. As we know the complete four-dimensional metric here, the related quantities can be read straightforwardly:

$$\alpha = \sqrt{H}, \ \beta^i = 0, \ \phi = H^{-1/6}, \tag{4.147}$$

and

$$K_{xx} = -\frac{\partial_t H}{2\alpha} = -\frac{k\pi A}{2}\frac{\cos[k\pi(x-t)]}{\{1 - A\sin[k\pi(x-t)]\}^{1/2}}, \quad K_{ij,\text{others}} = 0, \tag{4.148}$$

$$K = \gamma^{ij}K_{ij} = \frac{K_{xx}}{H} = -\frac{k\pi A}{2}\frac{\cos[k\pi(x-t)]}{\{1 - A\sin[k\pi(x-t)]\}^{3/2}}. \tag{4.149}$$

In Appendix C.2, we provide a detailed description of how this scenario is implemented as an initial condition in our evolution system. The static zero shift $\beta^i = \partial_t\beta^i = 0$ and the harmonic slicing $f(\alpha) = 1$ are adopted as the gauge conditions for this scenario.

The simulation test we report in this subsection utilizes a computational domain $\Omega = [-0.5, 0.5]^3$ on a regular grid with a periodic boundary condition. The periodic boundary condition is also a newly-added feature in ExaHyPE2. We divide the domain into 162 volumes per dimension, thus leading to a volume size of 0.006173. No AMR is enabled for this test. We set $A = 0.1$, $k = 2$ for the physical parameter. For the running parameters, we follow previous literature, setting $\kappa_1 = 0.1, \kappa_2 = 0, \kappa_3 = 0.5$ and $e = c = \tau = 1.0$. We use $\mu = 0.2$ for a smaller effect from the constraints as reported in Dumbser et al., 2018. As for the solver-specific parameters, the coefficient of the KO dissipation ϵ is set to be 8.0 and a CFL ratio $C = 0.1$ is adopted to be conservative. The parameters f and η are not related as the shift vector does not evolve in the gauge wave scenario.

The simulation runs to a code time of 10. The snapshots of $\tilde{\gamma}_{11}$ at four corresponding timestamps ($T = 3.25,\ 5.5,\ 7.75,\ 10.0$) are given in figure 4.5. The snapshot is an x-y plane slice of the simulation box perpendicular to the z-axis, with the colour map of the corresponding quantity. We also provide the profiles of $\tilde{\gamma}_{11}$ (in black) and $\tilde{A}_{11}$ (in red) along the x-axis at these four timestamps in figure 4.6, with their theoretical values plotted as well for comparisons.

The overall agreement of the evolution of the system with the theoretical prediction is good and the wave shows a constant and correct propagation speed. However, one may notice that there are deviations accumulating in the simulating patterns from the theoretical prediction over time. This can be seen more clearly in the temporal plots 4.7, which are generated utilising the static tracer module of Ex-

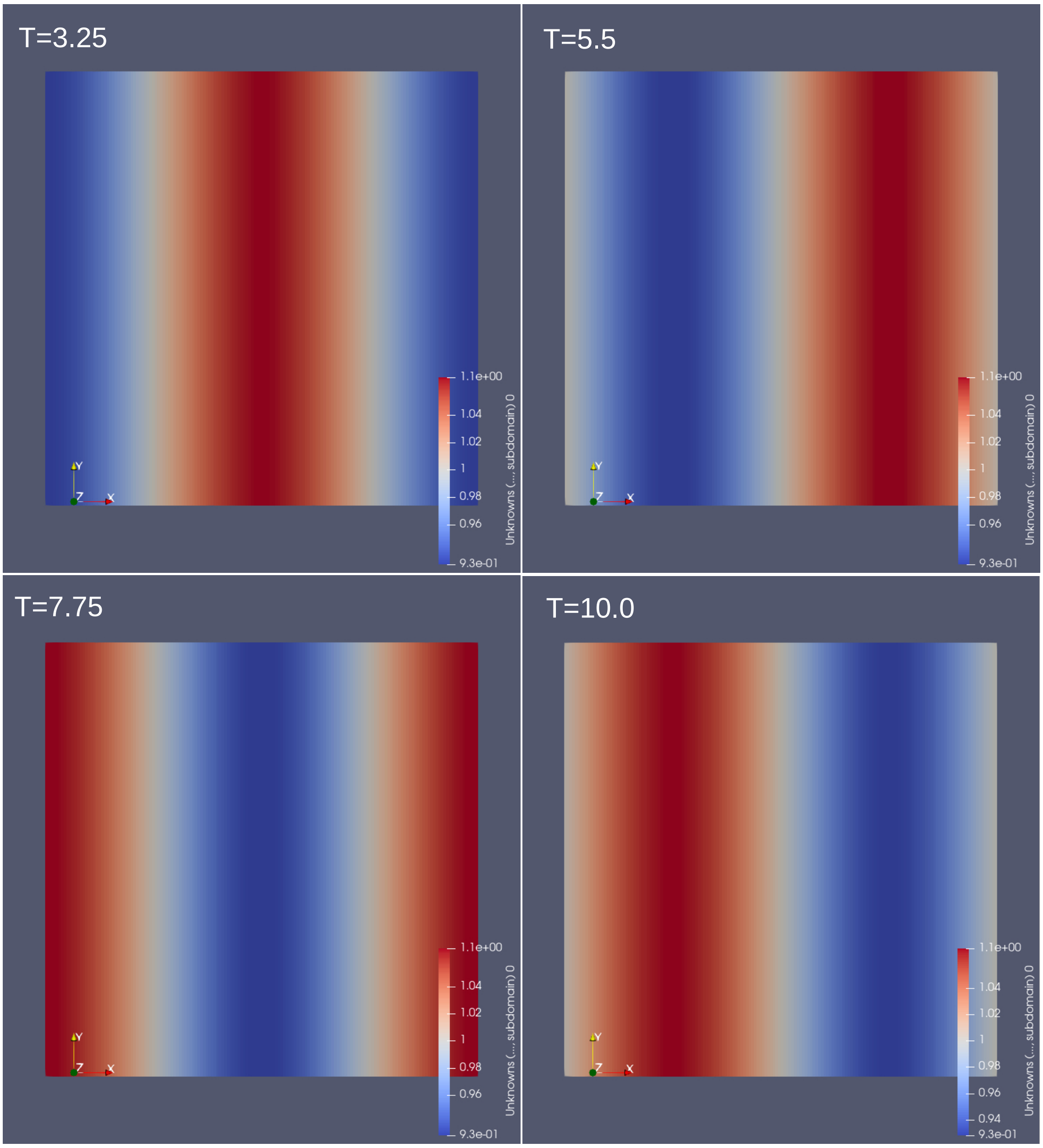

Figure 4.5: The snapshots of $\tilde{\gamma}_{11}$ at four different timestamps ($T = 3.25,\ 5.5,\ 7.75,\ 10.0$) in the gauge wave test. The snapshot shows the x-y plane slice of the computational domain at $z = 0$. A gauge wave with a velocity of one code unit travels in the right direction and re-enters the domain on the left due to the periodic boundary condition. The initial amplitude of this wave is around 0.06 as given in equation (C.2.18).

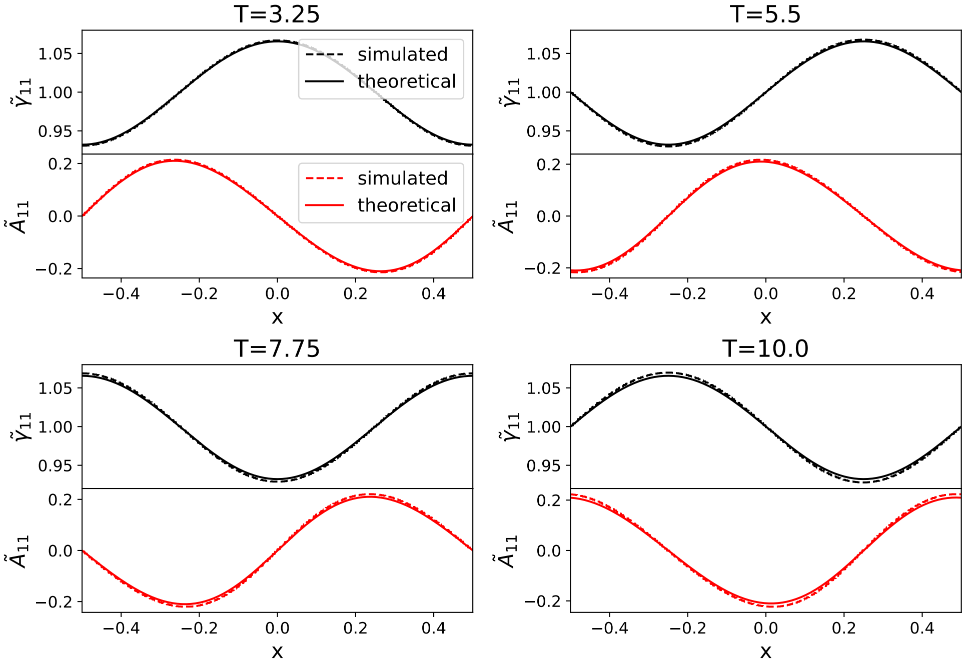

Figure 4.6: The profiles of $\tilde{\gamma}_{11}$ and $\tilde{A}_{11}$ at four different timestamps ($T = 3.25,\ 5.5,\ 7.75,\ 10.0$) in the gauge wave test. The profiles are extracted along the x-axis of the simulation domain. In each subfigure, the curves obtained in the simulation are plotted as dash lines in black and red respectively, while the analytical solutions of their counterpart are given in solid lines with the same colours. The exact formulation of the analytical solution can be found in the appendix, equations (C.2.18) and (C.2.21). The simulation result shows a general agreement with the theoretical prediction. However, a growing mismatch between them can be observed in the plots at later times.

aHyPE2. In the upper panel of the figure, we illustrate the evolution of the $\tilde{\gamma}_{11}$ (in black) and α (in red) at two different locations $[x, y, z] = [-0.1, 0, 0]$, $[0.1, 0, 0]$ respectively and compare with their theoretical profiles. Clearly, the amplitude of the propagating wave undergoes an unexpected growth over time, which should indicate the numerical error is accumulating in the evolution system.

This is further proved in the lower panel of figure 4.7, where deviations from the Hamiltonian constraint H (in blue) and the first component of the momentum constraint M_1 (in orange) are plotted against the code time at the corresponding tracer location $[-0.1, 0, 0]$. Both constraints are increasingly violated over time. The $Z4$ scheme, which is designed to propagate the constraint violation off the domain, seems not to work very well in our test. However, it is noted that our further tests have shown a numerical convergence in the gauge wave scenario, i.e., when we increase the resolution of the simulations, the constraint violation develops slower than it does in these low-resolution tests.

One possible theoretical explanation of the growing amplitude is that we are using periodic boundary conditions, where all quantities, thus also the numerical errors, will re-enter the system after they propagate off the domain. The magnitude of the KO dissipation and other damping components in the evolution system may also play a role here. Section 4.4 below will have more discussions on the long-term instabilities of our application.

4.3.2 Single Schwarzschild black hole

We now examine the performance of our application in simulating puncture black hole systems. In our first test, a single Schwarzschild black hole is considered. The black hole is placed at the origin of the three-dimensional domain, with an ADM mass $M = 1$ and zero spin $S = 0$. The domain in this test has a size of $[-9M, 9M]^3$ and three levels of AMR based on the radial distance from the central origin, which refines the domain at radius $r = 7M$ and again at $r = 3M$. As we adopt a three-partition strategy for refinement, the volume lengths of each level are $[0.333M, 0.111M, 0.037M]$. The Sommerfeld condition we introduced in section 2.3.2 is used to suppress the reflection effect from the boundary. To avoid resolving

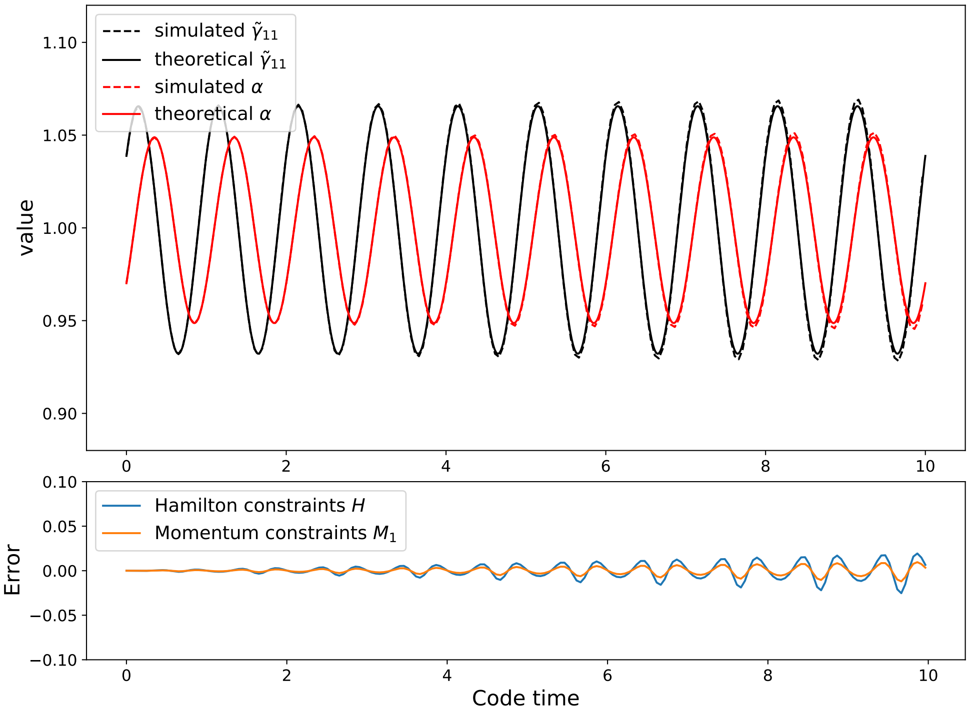

Figure 4.7: The temporal profiles of the quantities utilizing the tracer feature. *Upper panel*: The temporal profiles of $\tilde{\gamma}_{11}$ (in black) and α (in red), at the positions $[-0.1, 0, 0]$, $[0.1, 0, 0]$ respectively, are plotted in dashed lines. Their theoretical counterparts, which are given in equations (C.2.18) and (C.2.25), are illustrated in solid lines with the same colours. The two quantities show a constant phase difference due to their locations. A growing deviation of the simulation result from the theoretical profiles can be noticed. *Lower panel*: The temporal profiles of violations of the Hamiltonian constraint H and the first component of the momentum constraint M_1 at the location $[-0.1, 0, 0]$ are plotted in blue and orange lines respectively. While showing a fluctuation over time, the increasing amplitudes of both deviations indicate the numerical error is accumulating throughout evolution.

numerical infinity at the puncture location, no volume center coincides with the origin.

We construct the initial condition for this test utilizing the ported version of the TwoPuncture module from EINSTEINTOOLKIT library in our code and choose the average initial lapse:

$$\alpha = \frac{1}{2}\left(\frac{1-\frac{1}{2}\left(M/r\right)}{1+\frac{1}{2}\left(M/r\right)}+1\right). \tag{4.150}$$

Note that this is not the only possibility for the initial lapse, and other initial profiles are used in other numerical codes, e.g., GRCHOMBO, as well. However, we do not see a significant difference when using different shapes of initial lapse in our single black hole test. The initial shift vector is set to be zero initially. The extrinsic curvature K_{ij} also vanishes in the initial condition: its trace is zero as we assume the maximal slicing $K = 0$, and its residual is zero as no linear or angular momentum is presented in this system (see equations 4.40-4.41). Furthermore, we also assume conformal flatness in the initial condition, thus $\tilde{\gamma}_{ij} = 1$. Besides the lapse field α, the only non-trivial quantity in the initial condition of this single Schwarzschild black hole is the conformal factor ϕ, which can be solved analytically from equation (4.43) as its right-hand size now vanishes. The solution of it is

$$\phi \equiv \psi^{-2} = \left(1+\frac{M}{2r}\right)^{-2}. \tag{4.151}$$

We use the same setup and running parameters in the single black hole test as we did in the gauge wave simulation: $\kappa_1 = 0.1, \kappa_2 = 0, \kappa_3 = 0.5, e = c = \tau = 1.0$ as well as $\mu = 0.2$, for a similar evolving system as the original CCZ4 system. The fully functional gamma driver condition is employed for the evolution of the shift vector β^i in the black hole system, and we set the parameters as $f = 0.75$ and $\eta = 1$ in (4.73) and (4.79). The KO coefficient and the CFL ratio remain at 8 and 0.1, respectively.

The simulation runs until a code time of 46 when an instability terminates the code. The snapshots of the conformal factor ϕ at four timestamps ($T = 10,\ 20,\ 30,\ 40$) are given in figure 4.8, which again shows the slices cut along the x-y plane. One can observe a cross pattern showing up around the puncture in the late time of evolution. We also illustrate the profiles of four important quantities (ϕ, α, K and $\tilde{\gamma}_{11}$) along the x-axis in figure 4.9.

According to the simulation result, our application initially demonstrates a stable evolution of the black hole. However, instability appears after dozens of code times and continues to amplify. In the final stage of the test run, the solution deviates from the puncture shape significantly, eventually resulting in a crash. To show the issue more clearly, we plot the profile of Θ, which is the time component of the Z4 vector, along the x-axis through the evolution in figure 4.10. Only half of the central regime is illustrated as the profile is symmetric and we are mainly interested in its behaviour around the puncture. Profiles from six timestamps are plotted in total. While some initial damping is noticed (blue to orange curve), there is no clear propagation of the errors through the evolution, and the fluctuation of Θ (thus the constraint violations) remains nearly static in space.

We also investigate the source of those numerical errors in the central region. One possible explanation is that the derivatives of the auxiliary variables introduce a relatively large error during the evolution and make a significant impact, given that the primary quantities themselves are small around the punctures. For example, a primary quantity with a magnitude of 0.05 (a typical value for conformal factor) would have a derivative around unity under a volume size of 0.04, thus even a 1% numerical error arose in derivative evolution would have a comparable magnitude to the primary quantity itself. Therefore, we are reassessing our first-order formulation and examining if it has intrinsic issues that make a second-order re-formulation necessary. We again refer readers to the discussion section 4.4 below for details on this accumulating instability issues of our application when resolving the static system.

Another observation of this single black hole test is the persistent boundary effect. It is evident in figure 4.9 that an abnormal fluctuation appears on the two ends of the profiles, and some quantities (α, $\tilde{\gamma}_{11}$) are affected more severely than others. While the Sommerfeld boundary condition has already suppressed boundary reflections compared to the homogeneous Neumann condition, it still does not completely resolve the issue in this case in our tests. The main reason is that the

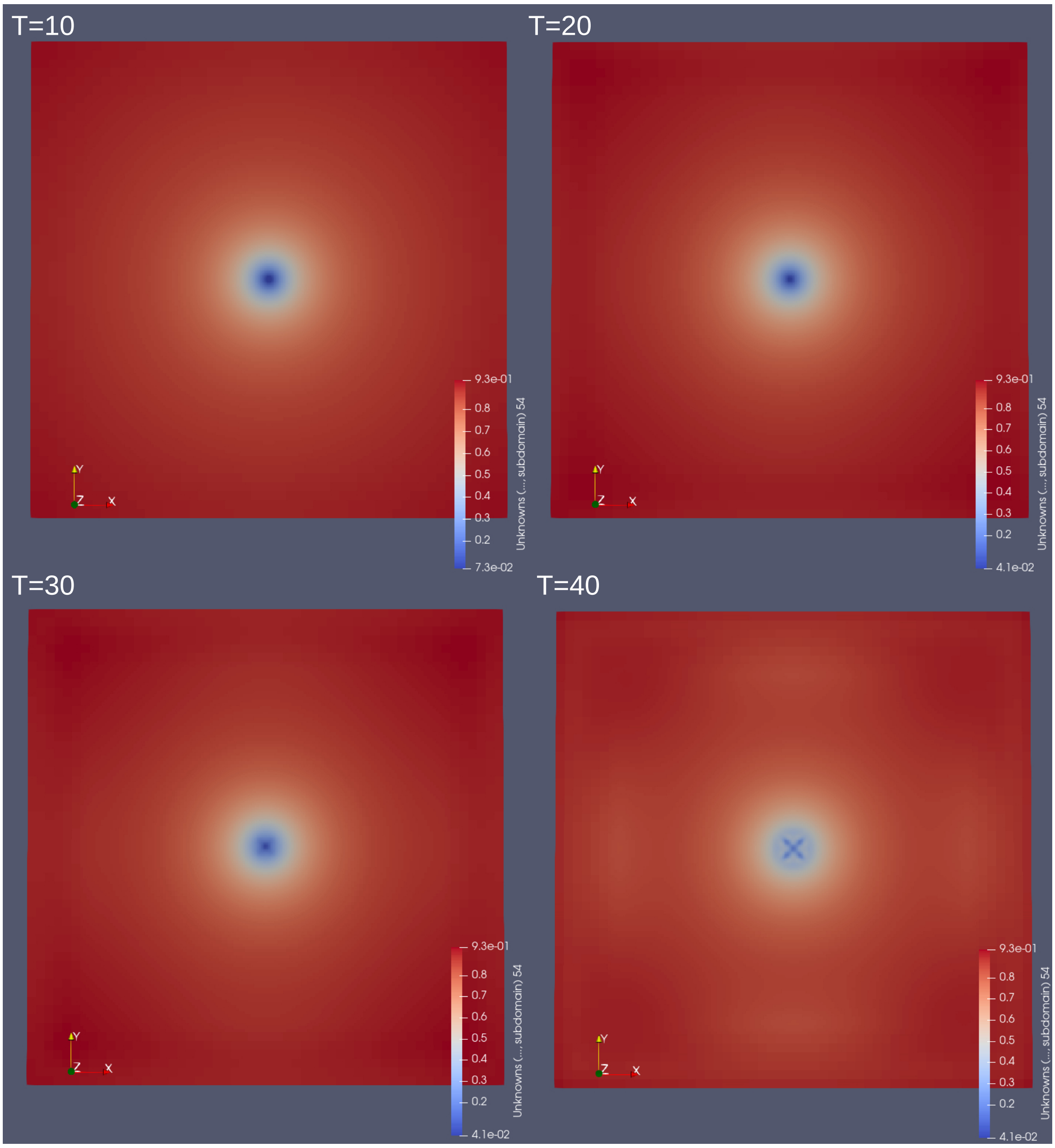

Figure 4.8: The snapshots of the conformal factor ϕ in the single Schwarzschild black hole test, at four different timestamps ($T = 10,\ 20,\ 30,\ 40$). The snapshot is given as the x-y plane slices through the computational domain where $z = 0$. The puncture representing the black hole is located at the origin, the center of the slices. The initial evolution of the system is relatively stable while instability appears at later times, and one can see clearly a cross pattern arises around the puncture in the last snapshot.

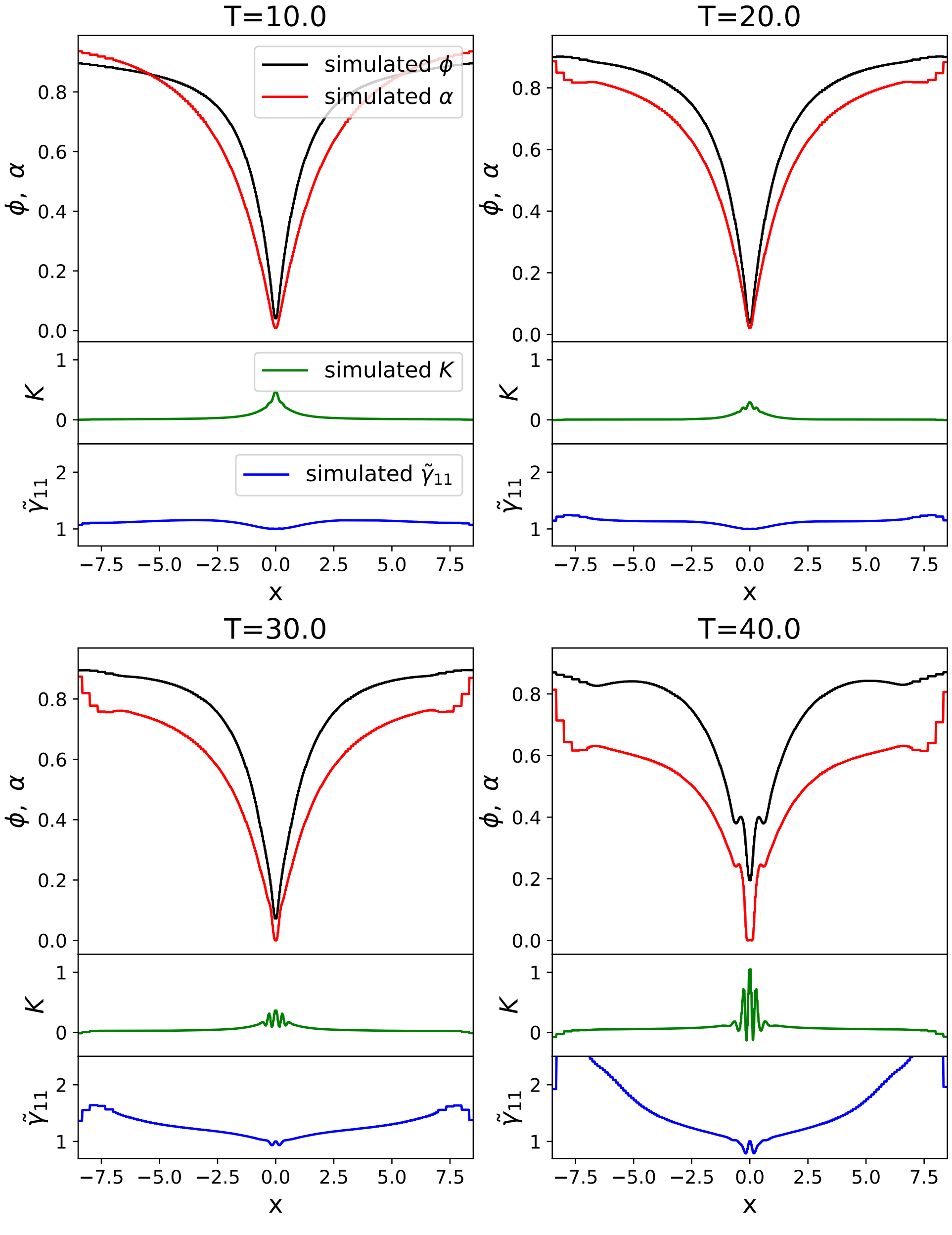

Figure 4.9: The profiles of four quantities (ϕ, α, K and $\tilde{\gamma}_{11}$) along the x-axis are illustrated at four timestamps ($T = 10,\ 20,\ 30,\ 40$). As the instability appears, ϕ and α show a discontinuity of the gradient, which later develops into bumps on the two sides of the punctures. K and $\tilde{\gamma}_{11}$ also show fluctuations in the central region at later times. Moreover, Those profiles exhibit fluctuations at the two ends, which arise from the compromised boundary condition we implemented. See the text for more details.

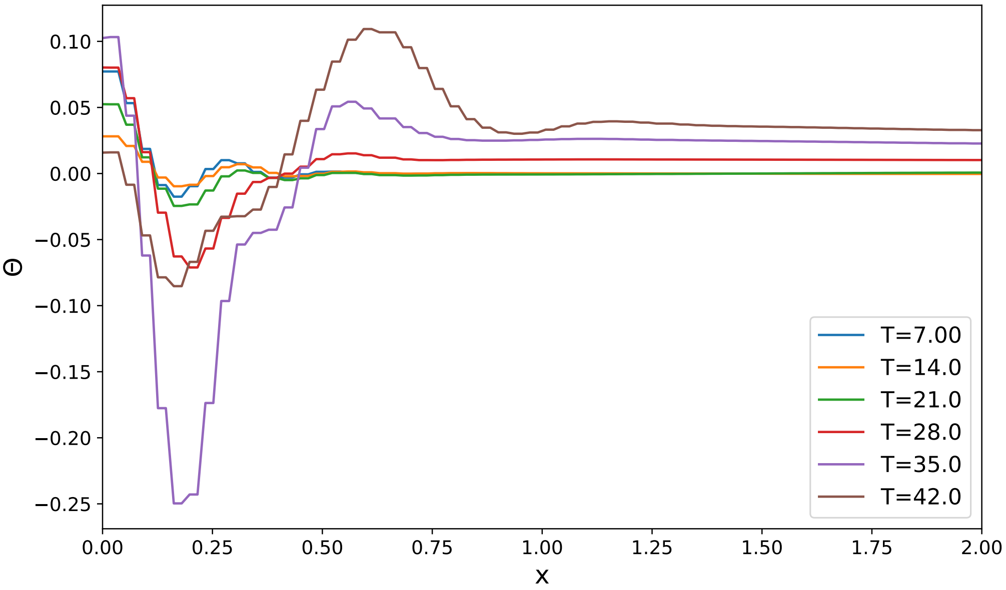

Figure 4.10: The profiles of Θ along the positive x-axis are given at six different timestamps ($T = 7,\ 14,\ 21,\ 28,\ 35,\ 42$) through the evolution, illustrating the behaviour of the time component of the Z vector. While some initial damping can be observed (blue to orange line), the magnitude of Θ increases over time, and no clear propagation of the error away from the puncture is evident in the evolution of the system. That may explain the accumulating numerical errors in our tests.

simulation box we use is too small and its boundary does not satisfy the assumptions of the Sommerfeld condition (spherical symmetry and wave-like solution), leading to a compromised performance. Although enlarging the simulation domain can potentially help to avoid this issue, we choose to continue with our current domain size as our primary focus is on solving the instability in the central region. Therefore, using a bigger domain would be an unnecessary waste of computational resources for now, but this will be considered in future.

4.3.3 Binary Black Hole Head-on Collision

In the following tests, we consider dynamic systems that contain binary black holes. The first scenario is the head-on collision of two black holes, which shows some essential features of the binary black hole system.

The two black holes are both set to have an ADM mass of $M_+ = M_- = 0.50$ and zero spin, $S_+ = S_- = 0$. The black holes are static initially, with a distance of $d = 4M$ between them. Without loss of generality, we locate them on the x-axis at the coordinates $[2M, 0, 0]$ and $[-2M, 0, 0]$. The domain of this test is similar to the one we used for the single black hole simulation, having a size of $[-9M, 9M]^3$ and three layers of AMR based on radial distances from the origin. The domain gets refined at radii $r = 7M$ and $r = 3M$ which gives volume length at each level as $[0.333M, 0.111M, 0.037M]$. The Sommerfeld boundary condition is applied and no volume centre coincides with the exact puncture location, as we did in the single black hole simulation to avoid numerical infinities.

The TwoPuncture module is again utilized to generate the initial condition for this scenario. Unlike the single black hole setup, the evolution system in this case does not have an analytical solution. As interactions between multiple sources of gravity are presented, the bare mass is different from the ADM mass as shown in equation (4.45), and this setup gives a bare mass of $m_+ = m_- \approx 0.47214$. While the numerical solution of the Hamiltonian constraints is handled by the external library, the initial lapse function still needs to be specified. We again use the average scheme of the lapse:

$$\alpha = \frac{1}{2}\left(\frac{1-\frac{1}{2}\left(m_-/r_-\right)-\frac{1}{2}\left(m_+/r_+\right)}{1+\frac{1}{2}\left(m_-/r_-\right)+\frac{1}{2}\left(m_+/r_+\right)}+1\right), \tag{4.152}$$

where r_+ and r_- are the distances from the considered volume to the two punctures respectively. The initial shift is zero. We keep the running parameters of the evolution system consistent with the single black hole case, thus $\kappa_1 = 0.1, \kappa_2 = 0, \kappa_3 = 0.5, e = c = \tau = 1.0$ and $\mu = 0.2$. The gamma driver condition is still employed with the parameters as $f = 0.75$ and $\eta = 1$. The KO coefficient and the CFL ratio are also left unchanged.

The simulation terminates at a code time of 35 where similar numerical errors to what we reported in the single black hole case ruin it. However, the collision of the binary black holes is correctly resolved, and the snapshots of the conformal factor ϕ through the dynamic process are shown in figure 4.11. We plot the snapshots for four timestamps ($T = 6,\ 12,\ 18,\ 24$) and show the x-y plane with the three-dimensional warped wireframes, where the punctures are illustrated as depressions. To further show the dynamic feature of the system, we illustrate the gauge quantities, the lapse α as the colour map and the shift β^i as the vector field in figure 4.12. Notice the reversed direction of the shift vector at the puncture location represents the moving direction of the punctures. In figure 4.13, we also provide the profiles of four important quantities (ϕ, α, Θ and $\tilde{\gamma}_{11}$) along the x-axis at the corresponding timestamps.

Compared with the instability arising in simulating static black hole systems, this head-on collision test yields a stable evolution of the acceleration, approaching and collision of the black holes. The constraint violations, which are represented by the Z vector, also do not exhibit unexpected growth throughout the dynamic process. It may be noticed that the magnitudes of Θ in this head-on collision test are actually larger than the ones observed in the single black hole simulation (see figure 4.10). However, the peaks of Θ always coincide with the punctures and also travel with them through the domain, and so the numerical errors associated with the constraint violations do not accumulate at specific locations. Therefore, the evolution is relatively stable, showing the Z4 scheme is working in this scenario. This is further supported by the fact that the head-on collision test crashes dozens

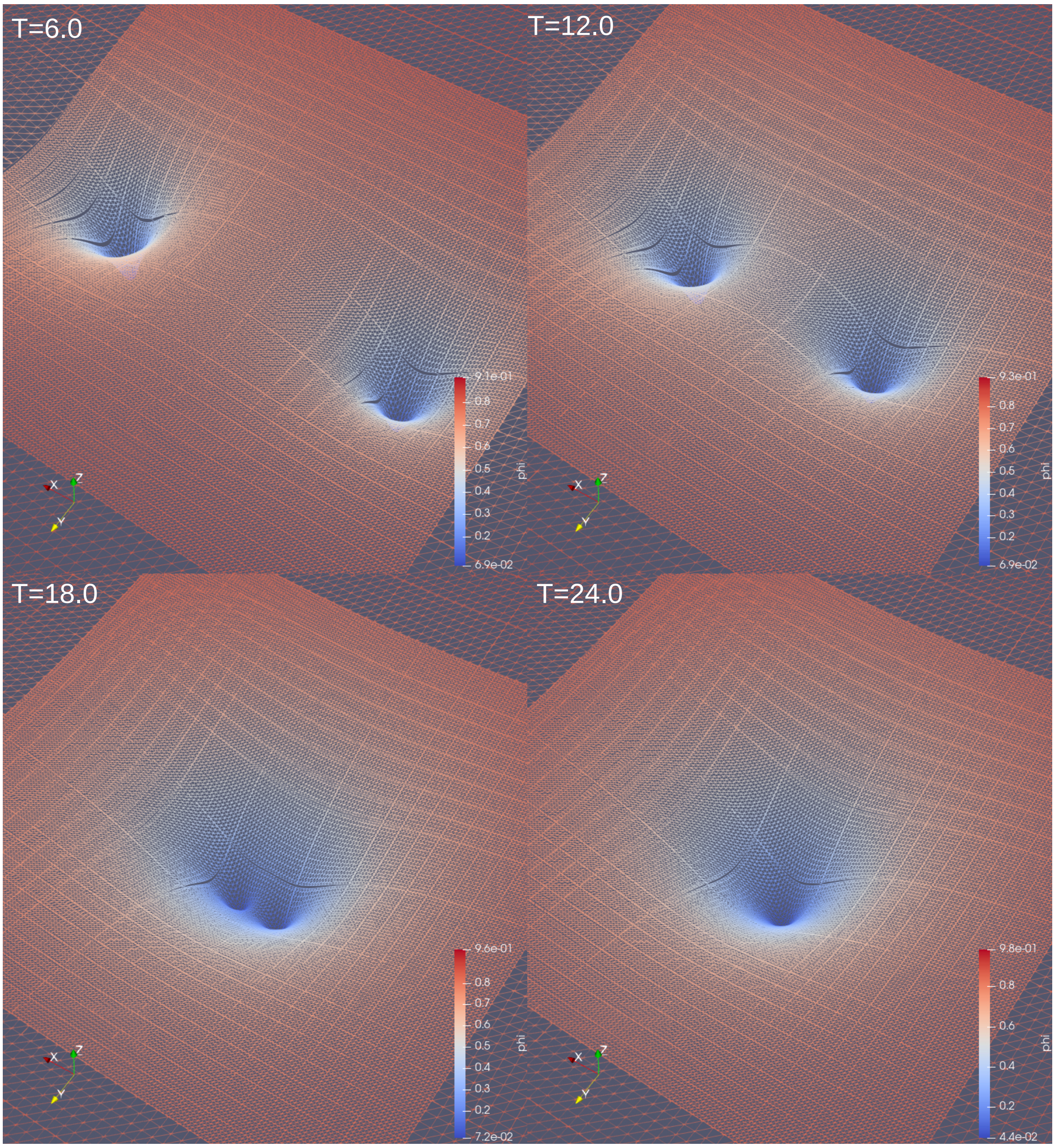

Figure 4.11: The snapshots of the conformal factor ϕ are plotted at four different timestamps ($T = 6,\ 12,\ 18,\ 24$) throughout the evolution. The x-y plane is illustrated in the figure as the three-dimensional warped wireframes, in which the punctures are represented by depressions. The process of puncture merge can be observed clearly in the figure.

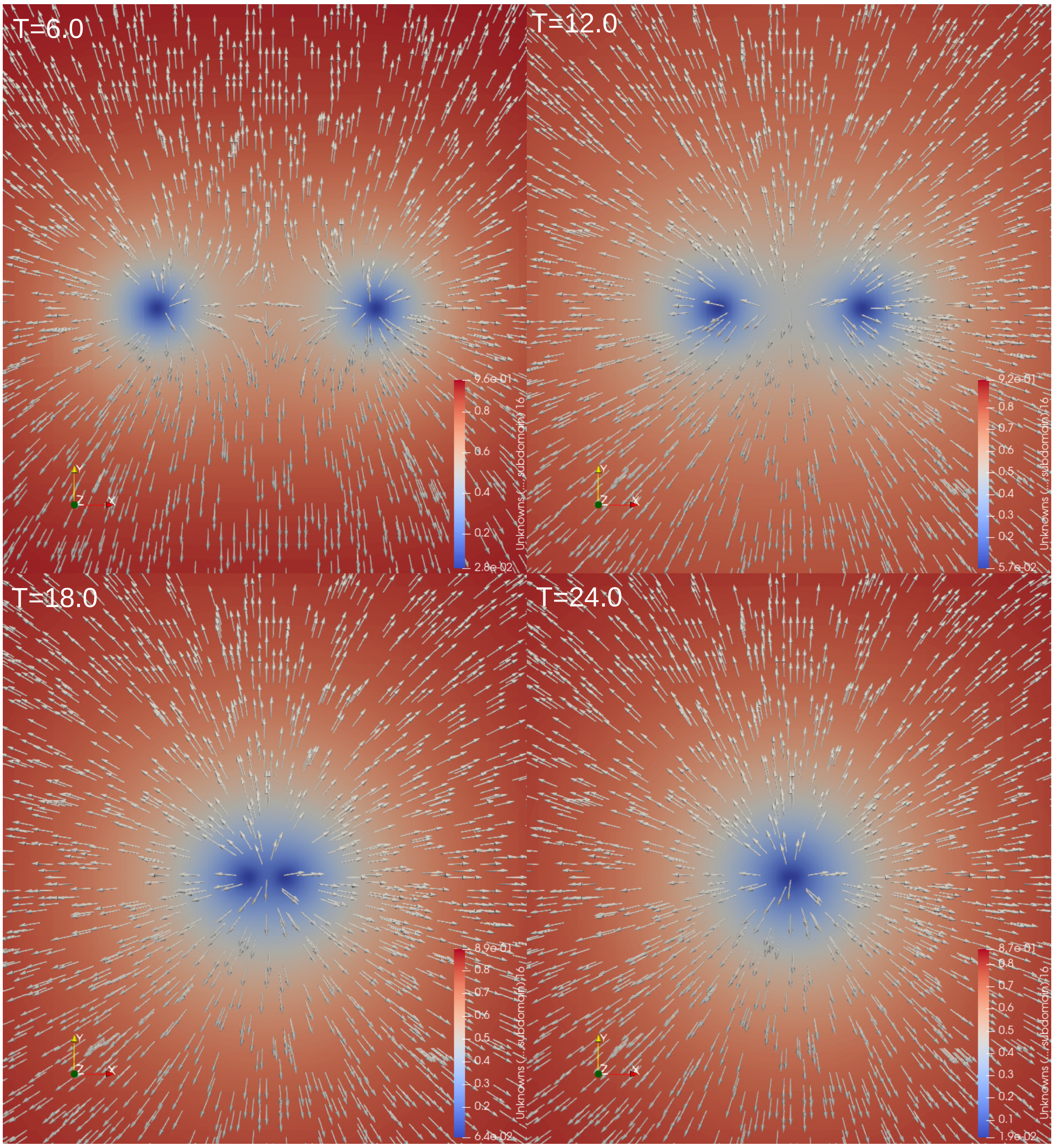

Figure 4.12: The snapshots of the gauge quantities, lapse α and shift β^i, are illustrated as the colour map and the vector field respectively, at four timestamps ($T = 6,\ 12,\ 18,\ 24$) throughout the evolution. The reversed direction of the shift vector at the puncture location indicates the moving direction of the puncture. For the head-on collision setup, the shift vector first starts from each puncture and then points outwards, opposite to the collision direction.

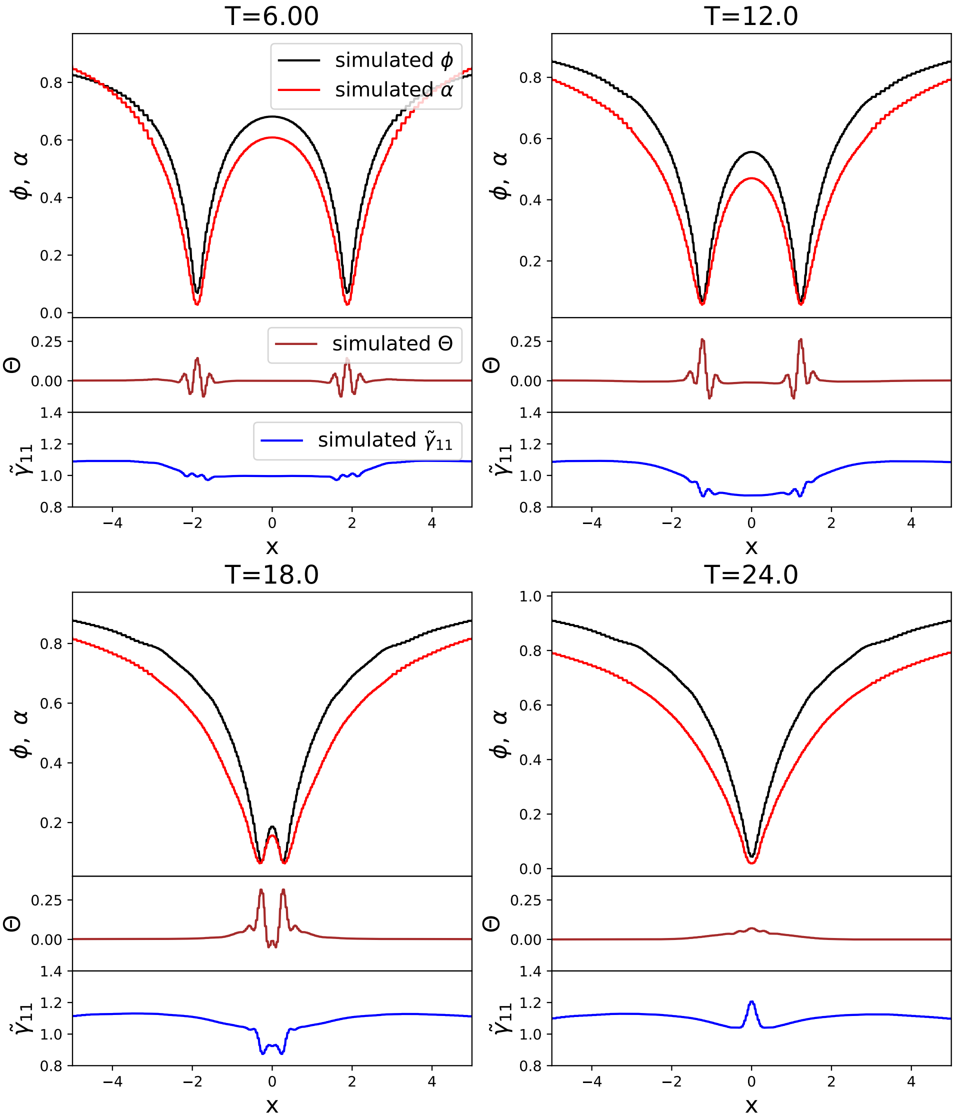

Figure 4.13: The profiles of four important quantities (ϕ, α, Θ and $\tilde{\gamma}_{11}$) are plotted along the x-axis at four timestamps ($T = 6,\ 12,\ 18,\ 24$) during the simulation. It clearly shows that the fluctuations in the component of the conformal metric and the constraint violations both coincide with the puncture locations and also travel with them throughout the evolution. The numerical errors thus get propagated and do not accumulate at certain locations.

of code times after the merge, where the simulation essentially reverts back to the single black hole scenario. Numerical errors then start to build up again and ruin the simulations. This test demonstrates that our application has the capability to perform simulations in dynamical binary black hole spacetimes, where the numerical errors are correctly propagated and cleaned up. We further explore this capability in the next test case.

4.3.4 Rotating Binary Black hole merger

In the final test cases, we consider the classical equal-mass rotating black hole merger to further examine the performance of our numerical relativity application, as well as our gravitational wave extraction code.

To construct the correct initial condition that allows for quasi-circular orbits, we utilize one set of the computed parameters from Tichy and Brügmann, 2004. In this setup, the binary black holes both have a bare (puncture) mass of $m_+ = m_- = 0.46477$ and zero spin, $S_+ = S_- = 0$. The initial distance between them is $d = 4M$. The black holes are located at the coordinates $[2M, 0, 0]$ and $[-2M, 0, 0]$ respectively, and rotate on the x-y plane, with the initial linear momentum $P^i_\pm = [0, \pm 0.19243M, 0]$. This setup corresponds to an ADM mass of $M = 0.5$ for each black hole and a total ADM mass of the system $M_{\text{tot}} = 0.98074$. We adopt a larger domain with a size of $[-12M, 12M]^3$ for our rotating binary black hole test with three levels of AMR. The domain gets refined at radius $r = 9M$ and again at $r = 5M$. The resolution of the simulation is set to be consistent with the head-on collision test, which gives the volume lengths at each level $[0.333M, 0.111M, 0.037M]$. The Sommerfeld boundary condition is imposed and the grid is examined such that no volume center coincides with the puncture position as we did for other black hole simulations. The calculator of the Newman-Penrose scalars ψ_4 as described in section 4.1.5 is also enabled to output ψ_4 in the domain.

We still use the TwoPuncture module to generate the exact initial condition and choose the average initial lapse

$$\alpha = \frac{1}{2}\left(\frac{1 - \frac{1}{2}(m_-/r_-) - \frac{1}{2}(m_+/r_+)}{1 + \frac{1}{2}(m_-/r_-) + \frac{1}{2}(m_+/r_+)} + 1\right), \tag{4.153}$$

with vanished initial shift. The running parameters remain the same as the ones in the head-on collision test, as follows: $\kappa_1 = 0.1, \kappa_2 = 0, \kappa_3 = 0.5, e = c = \tau = 1.0$ and $\mu = 0.2$. The gamma driver gauge condition uses the parameters $f = 0.75$ and $\eta = 1$. No modifications are made to the KO coefficient (8) and the CFL ratio (0.1), either.

The binary black holes complete two circular orbits before they merge at a code time of around 65. The simulation terminates at a code time of 67 when instability again crashes the evolution. The rotation and merging processes in the simulations are relatively stable. We plot the snapshots of the conformal factor ϕ and the real part of the ψ_4 quantities over the x-y plane in figure 4.14, at four different timestamps ($T = 15,\ 30,\ 45,\ 60$), plotted as three-dimensional wireframes and surfaces, respectively. We point out that the pattern of the ψ_4 quantity in the near field does not offer much useful information about the gravitational wave signal, as the wave extraction utilizing ψ_4 is only valid in the assumption of far-field. Nevertheless, an examination of its symmetrical behaviour can still be used to check whether our extraction is correctly implemented. We also present the gauge quantities α and β^i in figure 4.15 at these corresponding timestamps. These figures clearly show the rotating and merging evolution of the binary black holes.

We can also plot the trajectories of the punctures utilizing the moving tracers support of ExaHyPE2 as we introduced in section 2.5, and the result is illustrated in figure 4.16. The black holes finish one and a half circles before they start to merge at the center. The tracers are also used to measure the temporal behaviour of the ψ_4 quantity and figure 4.17 demonstrates the measures of ψ_4 at the location of four tracers $(x, y, z) = (0, 0, -8M), (0, 0, 8M), (8M, 8M, 0), (-8M, -8M, 0)$ throughout the evolution. The first two tracers are placed on the axis of symmetry, and the latter two are on the plane of the rotating black holes. No clear wave pattern can be observed, while the symmetry indicates that our wave extraction code is correctly implemented. Though we have all the code techniques required to calculate the mode quantities from the ψ_4 by spherical integral, we have not yet examined this feature due to the limitations in both code time and domain size.

Our application successfully performs this rotating binary black hole simulation

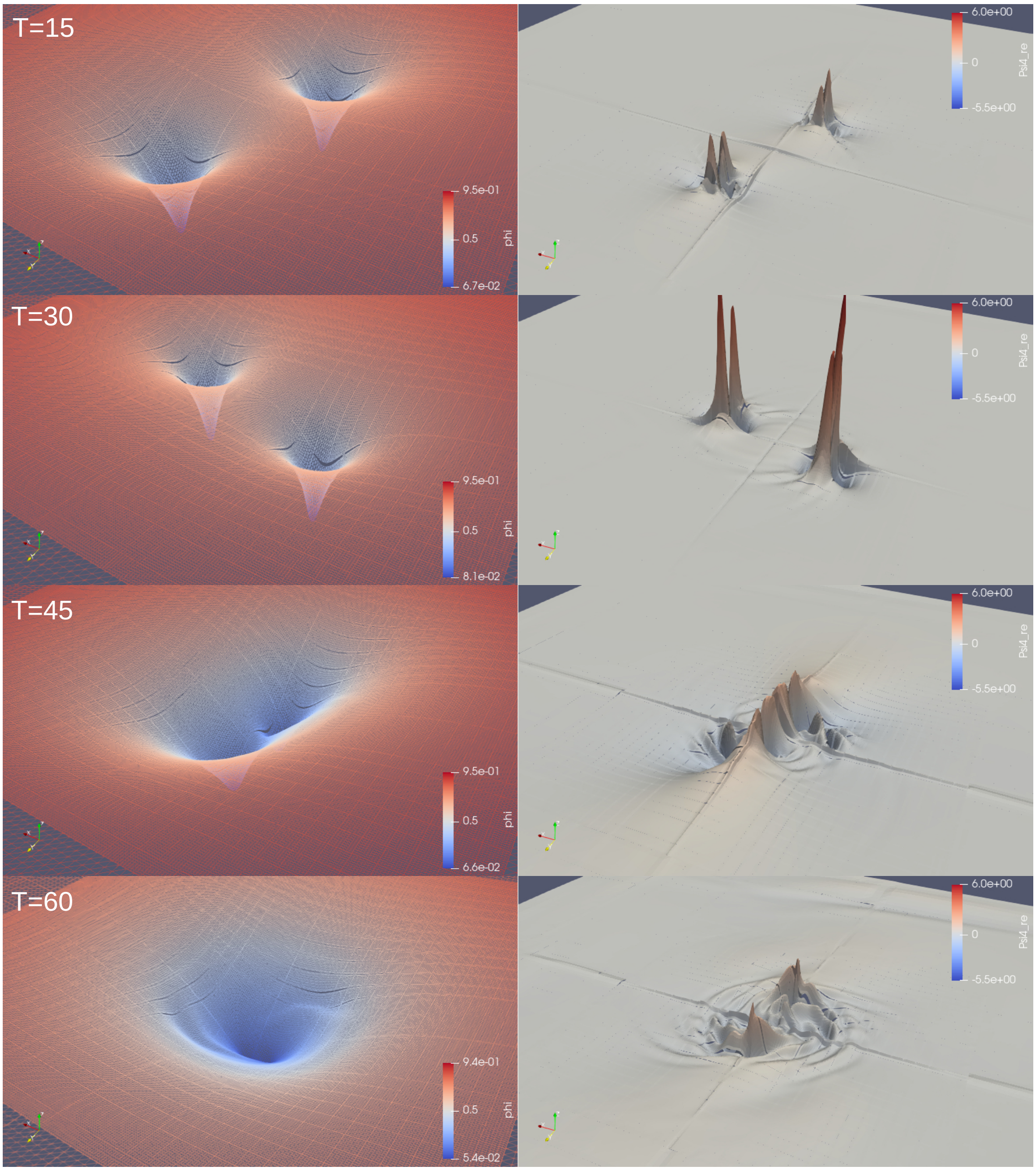

Figure 4.14: The snapshots of the conformal factor ϕ and the real part of the Newman-Penrose scalar ψ_4 are presented at four different timestamps ($T = 15,\ 30,\ 45,\ 60$), in three-dimensional warped wireframes and surfaces respectively. The rotation and merging of the black holes can be observed clearly. The pattern of ψ_4 in the near field does not provide much information about the gravitational wave, as the extraction is achieved utilizing a far-field approximation. However, its symmetrical shape serves as a useful indicator, confirming that our extraction was correctly implemented (also see figure 4.17 below).

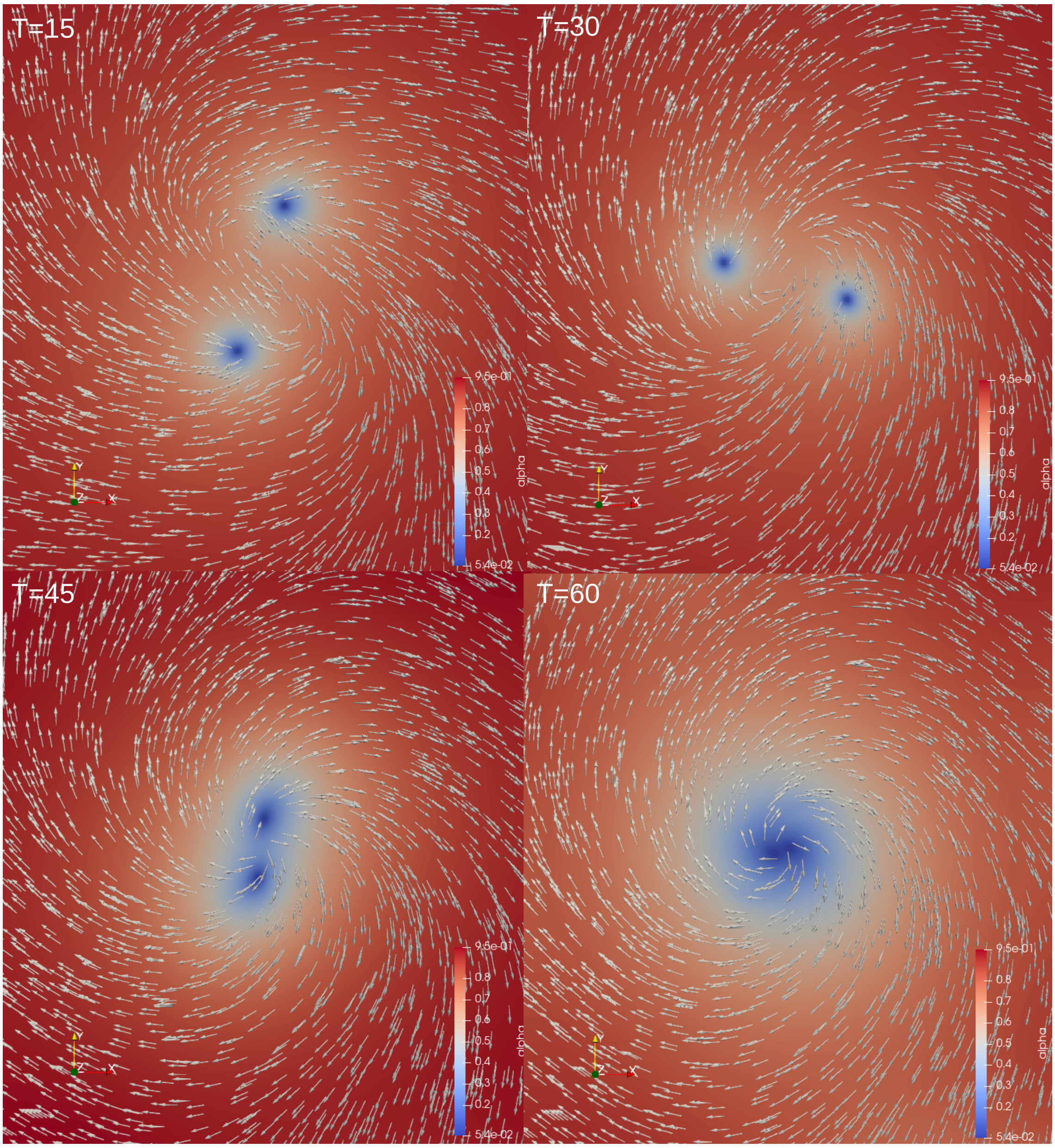

Figure 4.15: The snapshots of the gauge quantities, lapse α and shift vector β^i, are illustrated as the colour map and the vector field respectively, at four timestamps ($T = 15,\ 30,\ 45,\ 60$) throughout the evolution of the rotating binary black holes. As the punctures are rotating, the shift vector exhibits a spiral pattern and its reversed direction at the puncture location indicates the velocity direction of the puncture. The spiral pattern in the late stage also suggests that the remnant of the merged black holes carries spin.

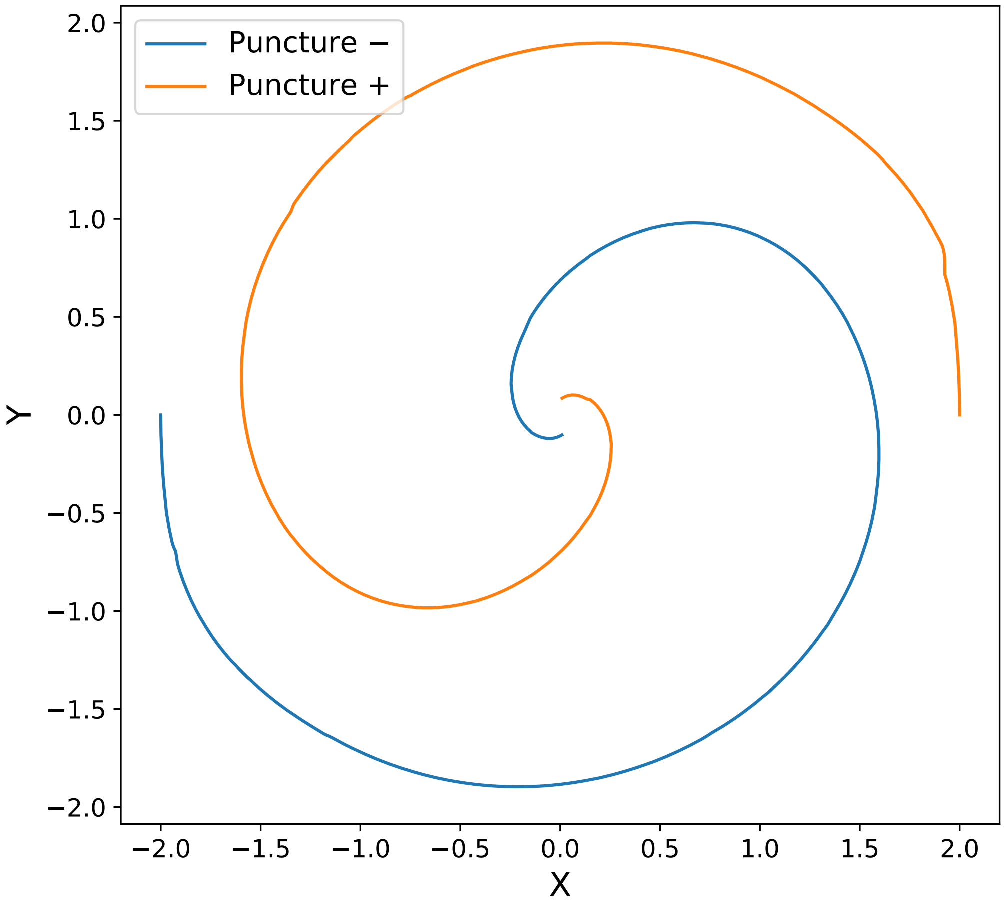

Figure 4.16: The trajectories of the two punctures in the binary black hole merger test. The two black holes ± are initially located at $[x, y, z] = [2M, 0, 0], [-2M, 0, 0]$ respectively, and their trajectories from the beginning to the crash of the code are plotting in the figure with orange and blue line. The black holes finish one and a half circles before they start to merge at the center. The initial distortion may come from a jump between different MPI regions, and we are still investigating this issue.

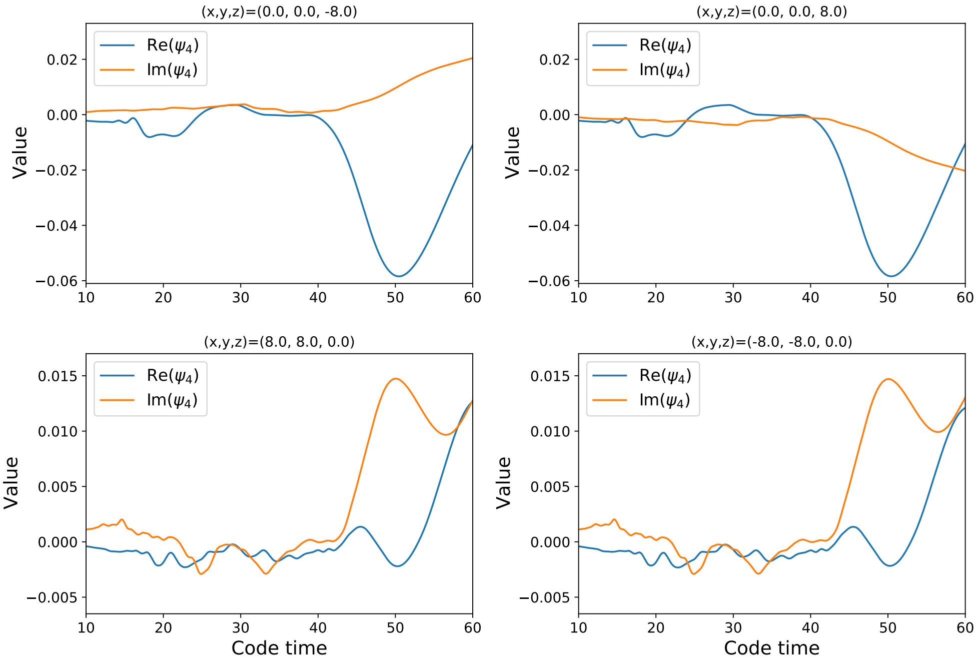

Figure 4.17: The temporal plot of the Newman-Penrose scalars ψ_4, at the location that is "relatively" far from the central gravity sources. Four tracers are placed at $(x, y, z) = (0, 0, -8M), (0, 0, 8M), (8M, 8M, 0), (-8M, -8M, 0)$ and the real and imaginary parts of calculated ψ_4 are given in subplots with blue and orange line respectively. Given the symmetry of the system, the two tracers on the axis of the symmetry have opposite real parts and identical imaginary parts, while the two on the rotation plane have both the same real and imaginary parts. No clear wave is observed as the limitation both in simulation time and domain size. The significant increase in the magnitudes shown after code time 40 is mainly due to the boundary effect, which pollutes the outskirts of the simulation domain. A much larger domain and longer simulation time is needed for a practical wave extraction in the future.

in general and has produced some preliminary results. The process of rotating and merging is accurately resolved in the simulation, and the locations of the punctures are correctly tracked by the (inverse) shift vector and our tracers. Once again, the evolution is quite stable, and an examination of the Θ field shows that the constraint violation is also travelling through the domain, thus avoiding the accumulation of numerical errors at specific locations as in the single black hole case. While the simulation before the merger is relatively successful, instability reappears in the evolution after the merge when the resulting black hole is roughly static. Consequently, numerical errors quickly build up and prevent us from resolving the ring-down stage of the evolution.

In summary, this result section shows the capability of our code in simulating various scenarios of the vacuum and black hole spacetimes. The tests, especially the binary black hole ones, give a stable evolution, particularly in dynamic systems where the numerical errors get propagated. They demonstrate that correct physics is implemented in the code, and the numerical FOCCZ4 scheme works. However, we observe instability in the long-term evolution of static systems and it leads to corrupted solutions and code crashes in our simulations. Therefore, resolving this issue is our primary focus in the current code development. We will delve further into this topic in the following discussion section.

4.4 Discussion

In this chapter, we have covered the theoretical background, code implementation and simulation tests of our numerical relativity application. As shown in the previous section, our code successfully conducts various simulations in vacuum and black hole spacetimes and provides stable evolution for a certain period of time. However, we have encountered instability issues in simulations of static systems, resulting in magnitude amplification (gauge wave) and the unexpected behaviour (single black hole) of quantities. The same issue also appears in our dynamic binary black hole system, which crashes our simulation after the black holes reach a nearly static state. We are thus unable to perform extended simulations for post-merge evolutions and remnants of the binary black hole systems.

Therefore, future code developments will focus on this instability issue. We have devoted significant effort to addressing this instability issue, and while a final solution is not yet reached, we have gained some ideas of the potential causes of the issue and possible ways to avoid them. The investigation of the parameter space in our tests indicates that the running parameters have little effect on this instability. While different choices of running parameters do influence the exact evolution patterns, the unexpected growth of the constraint violation is still observed in the domain during these tests. This instability suggests a more intrinsic issue in our code, which may arise from two aspects: our evolution formulation or our technical scheme.

The first potential cause of the instability in the evolving formulation is the approach we used in calculating derivatives. As we mentioned in the single black hole test, the instability shows its strongest effect around the puncture, where some of the primary quantities, such as α and ϕ, have a quite small value that is close to zero. On the other hand, the corresponding auxiliary variables (A_i and P_i) of them are relatively large due to the sharp shape of the puncture. Therefore, numerical errors arising from calculating the derivative of those auxiliary variables would have a strong impact on the primary variables themselves. To further examine this possibility, we adopt a variable replacement in some simulation as a test: instead of evolving the conformal factor ϕ in our system, we evolve its power function $\chi := \phi^n$, where n is an arbitrary integer that is larger than one[1]. A similar replacement is applied to its auxiliary variable as well: $P_i^\chi := \partial_i \chi = n\phi^{n-1}\partial_i \phi$. This replacement helps to suppress the issue we explain above by making the puncture well "shallower", thus reducing the magnitude of the auxiliary variables as well as the errors in their derivatives computations. The other quantities also need to be modified

---

[1] This operation is equivalent to rescale our conformal decomposition, and χ is our new conformal factor now. However, we will continue to treat ϕ as the conformal factor in the following content for the consistency of descriptions.

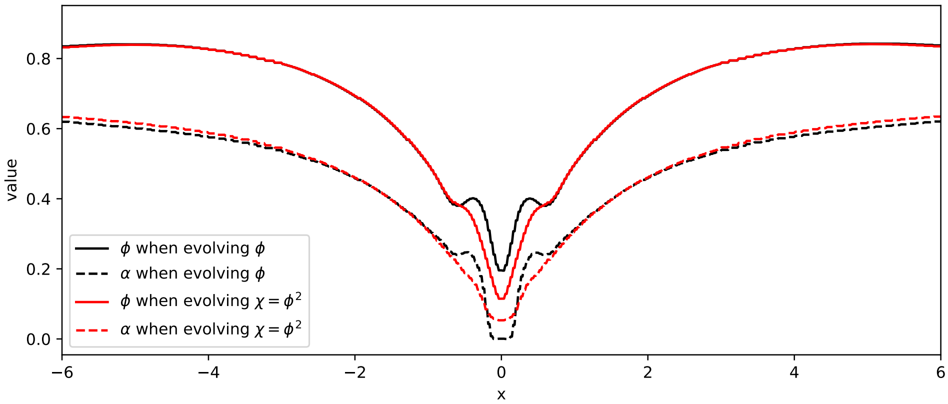

Figure 4.18: The comparison of the profiles of the conformal factor ϕ and lapse α along the x-axis in the single black hole scenario, at the same timestamp ($T = 40$). The profiles from the test with evolving $\phi (n = 1)$ are plotted in black solid and dashed lines, while the ones from the $\chi = \phi^2 (n = 2)$ case are given in red solid and dashed lines. It is clear that a higher power of ϕ in the evolution system suppresses the instability. However, we notice that a similar trend of bump has already developed for red lines as well, and the simulation in this $n = 2$ case also crashes after some time, though later than the $n = 1$ case.

accordingly, e.g., the modified Ricci tensor $R_{ij} + 2\nabla_{(i}Z_{j)}$ now have a formulation as

$$\begin{aligned}
R_{ij} + 2\nabla_{(i}Z_{j)} = \\
&- \frac{1}{2}\tilde{\gamma}^{kl}\partial_k\partial_l\tilde{\gamma}_{ij} + \tilde{\gamma}_{k(i}\partial_{j)}\hat{\Gamma}^k + \frac{1}{2}\hat{\Gamma}^k\partial_k\tilde{\gamma}_{ij} + \tilde{\gamma}^{lm}\left(\tilde{\Gamma}^k_{li}\tilde{\Gamma}_{jkm} + \tilde{\Gamma}^k_{lj}\tilde{\Gamma}_{ikm} + \tilde{\Gamma}^k_{im}\tilde{\Gamma}_{klj}\right) \\
&+ \frac{1}{n}\chi^{-1}\left(\tilde{\nabla}_i\tilde{\nabla}_j\chi + \tilde{\gamma}_{ij}\tilde{\gamma}^{kl}\tilde{\nabla}_k\tilde{\nabla}_l\chi\right) + \frac{1}{\phi^2}\frac{2}{n}\chi^{-1}Z^k\left(\tilde{\gamma}_{ik}\partial_j\chi + \tilde{\gamma}_{jk}\partial_i\chi - \tilde{\gamma}_{ij}\partial_k\chi\right) \\
&- \frac{1}{n^2}\chi^{-2}\left[(n-1)\partial_i\chi\partial_j\chi + (n+1)\tilde{\gamma}_{ij}\tilde{\gamma}^{kl}\partial_k\chi\partial_l\chi\right]. \qquad (4.154)
\end{aligned}$$

It goes back to equation (4.95) when $n = 1$.

A comparison of the profiles of α and ϕ along the x-axis in the single black hole scenario with $n = 1$ and $n = 2$ is given in figure 4.18, at the same timestamp ($T = 40$). All other parameters are kept identical. It can be seen clearly that a higher power of ϕ helps to suppress the instability, not only in the conformal factor itself but also in the lapse field. However, the replacement does not address the issue completely, as a similar trend of bump appears in the profiles of the $n = 2$ case as well. It is further proved as the simulation with $n = 2$ runs longer than the $n = 1$ case but still crashes at a code time of 50.5.

We thus consider the possibility that our first-order formulation may suffer from an intrinsic issue of the so-called "two-step" scheme, where second-order derivatives are computed as the derivatives of the first derivatives. As we are using the finite difference scheme, this approach actually yields a different, and often larger, stencil compared to directly calculating the second derivatives. This scheme has the potential to introduce extra numerical errors, and the large stencil may also affect the behaviour of solutions around discontinuities in the domain, such as punctures. As a potential solution to the instability issue, we are actively developing and exploring the implementation of the second-order formulation of the CCZ4 system in our code. We hope to report the influence of this approach in future work.

Another possible source of the instability can be the implemented computation solver. We are currently using a fourth-order finite difference solver with the Euler time integration scheme, which may not be sufficient enough to resolve the complex numerical relativity system. The generated numerical error exceeds the limit that the dissipation scheme can handle and results in the amplifying magnitude we observe in the gauge wave test. On the other hand, numerical solvers with high order are likely to become unstable when resolving shock and discontinuity. Therefore a low-order solver may need to be added to the systems involving punctures. The low-order solver only works on the small regions around punctures and couples with the main high-order solver, which is responsible for solving PDE in the rest of the domain. This solver-coupling idea is utilized in the preliminary work of implementing the FOCCZ4 system on ExaHyPE1 (Dumbser et al., 2018). The authors implemented a high-order Ader-DG solver for most of the domain but employed a finite volume solver close to the puncture as a limiter. Our application may benefit from the same idea. As we introduced in chapter 2, new advanced solvers and the code framework for solver coupling are currently being worked on and we shall soon be able to examine them in our numerical relativity code.

Besides considering the source of the numerical errors, we also need to re-examine the Z4 scheme in our formulation. Currently, the scheme for propagating constraint violations appears to function well in the dynamic system in our application, which helps to yield stable evolutions there even with the persistent numerical errors. However, we observe no clear propagation when resolving the static system, especially in the single black hole system. Understanding its behaviour in both static and dynamic systems requires further investigation, and we plan to look into it more closely in subsequent research.

4.5 One Step Further: Evolution System in Scalar Field Theory

All the theories, formulations and results we presented above are based on standard General Relativity. One natural topic from there is whether we can extend our simulation to more general gravity theories. As we are still in the development stage for the code of standard gravity, we have not tested this possibility of extension at the code level yet. However, we have looked into the theory and formulation of a scalar-tensor field suitable for numerical evolution. We give a description of them in this section for completeness.

The formulation of an extended Einstein Hilbert action with a scalar field that self-interacts and couples to curvature is given as (Uzan, 2011):

$$\mathrm{S}{=}\int \frac{\mathrm{d}^4x}{16\pi G_*}\sqrt{-g}\left[F(\varphi)^{(4)}R - g^{ab}Z(\varphi)\partial_a\varphi\partial_b\varphi - 2U(\varphi) + 2g^{ab}\nabla_a Z_b\right] + S_{\text{matter}}\ [\psi; g_{ab}]\,. \tag{4.155}$$

We have again added the Z vector above and φ is the scalar field. $F(\varphi), Z(\varphi), U(\varphi)$ are three functions that only depend on the scalar field and determine how it couples with other elements in the spacetime. To be more precise, $F(\varphi)$ describes the coupling of the scalar field to curvature, $Z(\varphi)$ describes possible non-standard kinetic energy of the field, and $U(\varphi)$ describes the scalar field's self-interacting potential $S_{\text{matter}}\ [\psi; g_{ab}]$ is the action for the matter in the universe and we will focus on the vacuum solution and so will not include it in the following content. The variation of the action above gives the new extended field equations for metric and the scalar field:

$$\begin{aligned} F(\varphi)[{}^{(4)}R_{ab} - \frac{1}{2}g_{ab}{}^{(4)}R] + \nabla_a Z_b + \nabla_b Z_a - g_{ab}\nabla^c Z_c - \kappa_1[n_a Z_b + n_b Z_a + \kappa_2 g_{ab} n_c Z^c] = \\ 8\pi T_{ab} + Z(\varphi)[\nabla_a\varphi\nabla_b\varphi - \frac{1}{2}g_{ab}\nabla^c\varphi\nabla_c\varphi] + \nabla_a\nabla_b F(\varphi) - g_{ab}\nabla_c\nabla^c F(\varphi) - g_{ab}U(\varphi), \end{aligned} \tag{4.156}$$

$$2Z(\varphi)\nabla_a\nabla^a\varphi = -\frac{dF}{d\varphi}{}^{(4)}R - \frac{dZ}{d\varphi}\left(\partial_a\varphi\right)^2 + 2\frac{dU}{d\varphi}. \tag{4.157}$$

The trace-reversed field equation is

$$F(\varphi)^{(4)}R_{ab} + \nabla_a Z_b + \nabla_b Z_a - \kappa_1[n_a Z_b + n_b Z_a - (1+\kappa_2)g_{ab}n_c Z^c] = \\ 8\pi(T_{ab} - \frac{1}{2}g_{ab}T) + Z(\varphi)\nabla_a\varphi\nabla_b\varphi + \nabla_a\nabla_b F(\varphi) + \frac{1}{2}g_{ab}\nabla^c\nabla_c F(\varphi) + g_{ab}U(\varphi). \tag{4.158}$$

The Hamiltonian and momentum constraints in this formulation can be derived using a similar approach to the derivation of (4.17) and (4.18), and they give

$$\begin{aligned} F(\varphi)[R + K^2 - K_{ab}K^{ab}] = 16\pi c &+ Z(\varphi)[2K_\varphi^2 + \nabla_a\varphi\nabla^a\varphi] \\ &+ 2n^a n^b \nabla_a\nabla_b F(\varphi) + 2\nabla_a\nabla^a F(\varphi) + 2U(\varphi) \\ &+ 2\kappa_1[2\Theta + \kappa_2\Theta] - 2n^a n^b\nabla_a Z_b - 2n^a n^b \nabla_b Z_a - 2\nabla^c Z_c, \end{aligned} \tag{4.159}$$

$$\begin{aligned} F(\varphi)(D_b K^b_c - D_c K) = 8\pi S_c &+ Z(\varphi)K_\varphi \gamma^a_c \nabla_a\varphi - n^a\gamma^b_c\nabla_a\nabla_b F(\varphi) \\ &+ \gamma^a_c n^b \nabla_a Z_b + \gamma^a_c n^b\nabla_b Z_a + \kappa_1\gamma^a_c Z_a. \end{aligned} \tag{4.160}$$

In the equation above we defined the extrinsic curvature counterpart of the scalar field:

$$K_\varphi = -\mathcal{L}_n\varphi = -n^a\nabla_a\varphi \tag{4.161}$$

The following 3-dimensional evolution equations for a Z4 system can be derived from the field equations

$$\frac{1}{\alpha}(\partial_t - \mathcal{L}_\beta)\gamma_{ij} = \mathcal{L}_n\gamma_{ij} = -2K_{ij}, \tag{4.162}$$

$$\frac{1}{\alpha}(\partial_t - \mathcal{L}_\beta)\varphi = \mathcal{L}_n\varphi = -K_\varphi, \tag{4.163}$$

$$
\begin{aligned}
&\frac{1}{\alpha}(\partial_t - \mathcal{L}_\beta)K_{ij} = \mathcal{L}_n K_{ij} = \\
&\quad - \frac{1}{\alpha}\nabla_i\nabla_j\alpha - 2K_i^l K_{jl} + R_{ij} + KK_{ij} - \frac{1}{F(\varphi)}8\pi\left(S_{ij} - \frac{1}{2}(S-\rho)\gamma_{ij}\right) \\
&\quad + \frac{1}{F(\varphi)}\left[\nabla_i Z_j + \nabla_j Z_i - 2\Theta K_{ij} - \kappa_1(1+\kappa_2)\Theta\gamma_{ij} - U(\varphi)\gamma_{ij}\right] \\
&\quad - \frac{F''(\varphi) + Z(\varphi)}{F(\varphi)}\nabla_i\varphi\nabla_j\varphi - \frac{F'(\varphi)}{F(\varphi)}(\nabla_i\nabla_j\varphi - K_\varphi K_{ij}) \\
&\quad - \frac{1}{2}\gamma_{ij}\left[\frac{F''(\varphi)}{F(\varphi)}(\nabla_k\varphi\nabla^k\varphi - K_\varphi^2) + \frac{F'(\varphi)}{F(\varphi)}(\nabla^k\nabla_k\varphi + \nabla^k\ln\alpha\nabla_k\varphi - KK_\varphi)\right] \\
&\quad - \frac{1}{2}\gamma_{ij}\frac{F'(\varphi)}{F(\varphi)}\mathcal{L}_n K_\varphi,
\end{aligned}
\tag{4.164}
$$

$$
\begin{aligned}
&\frac{1}{\alpha}(\partial_t - \mathcal{L}_\beta)K_\varphi = \mathcal{L}_n K_\varphi = \left[2Z(\varphi) + 3\frac{F'(\varphi)^2}{F(\varphi)}\right]^{-1} \\
&\quad \left\{F'(\varphi)[R + K^2 - K_{ij}K^{ij}] + \left[Z'(\varphi) + 3\frac{F'(\varphi)F''(\varphi)}{F(\varphi)}\right]K_\varphi^2 - \left[2Z(\varphi) + 5\frac{F'(\varphi)^2}{F(\varphi)}\right]\nabla_i\nabla^i\varphi\right. \\
&\quad - \left[Z'(\varphi) + 2\frac{F'(\varphi)Z(\varphi)}{F(\varphi)} + 5\frac{F'(\varphi)F''(\varphi)}{F(\varphi)}\right]\nabla_i\varphi\nabla^i\varphi - \left[2Z(\varphi) + 3\frac{F'(\varphi)^2}{F(\varphi)}\right]\nabla_i\ln\alpha\nabla^i\varphi \\
&\quad + \left[2Z(\varphi) + 5\frac{F'(\varphi)^2}{F(\varphi)}\right]KK_\varphi + 2\frac{F'(\varphi)}{F(\varphi)}\left[4\pi(S - 3\rho) - 3\kappa_1(1+\kappa_2)\Theta + 2\nabla_i Z^i - 2\Theta K\right] \\
&\quad \left. + 2U'(\varphi) - 6\frac{F'(\varphi)}{F(\varphi)}U(\varphi)\right\},
\end{aligned}
\tag{4.165}
$$

$$
\begin{aligned}
&\frac{1}{\alpha}(\partial_t - \mathcal{L}_\beta)\Theta = \frac{1}{2}F(\varphi)[R + K^2 - K_{ij}K^{ij}] + \nabla_i Z^i - \Theta K - Z^i\nabla_i\ln\alpha - \kappa_1(2+\kappa_2)\Theta - 8\pi\rho \\
&\quad - \frac{1}{2}Z(\varphi)K_\varphi^2 - \left[\frac{1}{2}Z(\varphi) + F''(\varphi)\right]\nabla^i\varphi\nabla_i\varphi - F'(\varphi)\nabla_i\nabla^i\varphi + F'(\varphi)KK_\varphi - U(\varphi),
\end{aligned}
\tag{4.166}
$$

$$
\begin{aligned}
&\frac{1}{\alpha}(\partial_t - \mathcal{L}_\beta)Z_i = F(\varphi)(\nabla_j K_i^j - \nabla_i K) - 8\pi S_i - \kappa_1 Z_i + \nabla_i\Theta - 2K_i^j Z_j - \Theta\nabla_i\ln\alpha \\
&\quad - [Z(\varphi) + F''(\varphi)]K_\varphi\nabla_i\varphi - F'(\varphi)\nabla_i K_\varphi + F'(\varphi)K_i^j\nabla_j\varphi.
\end{aligned}
\tag{4.167}
$$

The detailed derivation can be found in Appendix B.2.1. Two extra evolution equations of φ and K_φ enter the system. Most notation follows those in section 4.1.3, and (double) prime indicates (second) derivatives with respect to φ: $F'(\varphi) = \mathrm{d}F(\varphi)/\mathrm{d}\varphi$, $F''(\varphi) = \mathrm{d}^2F(\varphi)/\mathrm{d}\varphi^2$. One notable observation is that now the evolution of extrinsic curvature is coupled with the evolution of K_φ as the red term we marked in equation (4.164).

We then continue to develop this formulation by applying conformal decomposition and adding auxiliary variables as described in section 4.1.3. A new auxiliary variable that represents the derivative of scalar field φ is also needed:

$$\Pi_i := D_i\varphi = \partial_i\varphi. \tag{4.168}$$

The detailed derivation (see Appendix B.2.2) finally leads us to the FOCCZ4 system evolving with a scalar field, containing 11 equations of primary variables:

$$\begin{aligned}\partial_t\tilde{\gamma}_{ij} &= 2\beta^k D_{kij} + \tilde{\gamma}_{ki}B^k_j + \tilde{\gamma}_{kj}B^k_i - \frac{2}{3}\tilde{\gamma}_{ij}B^k_k - 2\alpha(\tilde{A}_{ij} - \frac{1}{3}\tilde{\gamma}_{ij}\mathrm{tr}\tilde{\mathrm{A}}) \\ &\quad - \tau^{-1}(\tilde{\gamma} - 1)\tilde{\gamma}_{ij},\end{aligned} \tag{4.169}$$

$$\partial_t\phi = \beta^k P_k + \frac{1}{3}\phi\left(\alpha K - B^l_l\right), \tag{4.170}$$

$$\partial_t\varphi = \beta^k\Pi_k - \alpha K_\varphi, \tag{4.171}$$

$$\partial_t\alpha = \beta^k A_k - \alpha^2 f(\alpha)\left(K - K_0 - 2c\Theta\right), \tag{4.172}$$

$$\partial_t\beta^i = \beta^k B^i_k + f b^i, \tag{4.173}$$

$$\partial_t b^i - \beta^k\partial_k b^i = \partial_t\hat{\Gamma}^i - \beta^k\partial_k\hat{\Gamma}^i - \eta b^i, \tag{4.174}$$

$$\begin{aligned}&\partial_t\tilde{A}_{ij} - \beta^k\partial_k\tilde{A}_{ij} - \phi^2\Big[- D_iD_j\alpha + \alpha R_{ij} + \frac{\alpha}{F(\varphi)}(D_iZ_j + D_jZ_i - F'(\varphi)D_iD_j\varphi)\Big]^{TF} \\ &\quad = \tilde{A}_{ki}B^k_j + \tilde{A}_{kj}B^k_i - \frac{2}{3}\tilde{A}_{ij}B^k_k - 2\alpha\tilde{A}_{il}\tilde{A}^l_j + \alpha\tilde{A}_{ij}\Big[K - \frac{2\Theta}{F(\varphi)}\Big] \\ &\qquad - \phi^2\alpha\frac{F''(\varphi) + Z(\varphi)}{F(\varphi)}[\Pi_i\Pi_j]^{TF} + \frac{F'(\varphi)}{F(\varphi)}\alpha K_\varphi\tilde{A}_{ij} - \tau^{-1}\tilde{\gamma}_{ij}\mathrm{tr}\tilde{\mathrm{A}},\end{aligned} \tag{4.175}$$

$$\begin{aligned}&\partial_t K - \beta^i\partial_i K + D^iD_i\alpha - \alpha\Big[R + \frac{2}{F(\varphi)}D^iZ_i - \frac{5F'(\varphi)}{2F(\varphi)}D^iD_i\varphi\Big] \\ &\quad = \alpha K\Big(K - \frac{2\Theta}{F(\varphi)}\Big) - \frac{3}{F(\varphi)}\alpha\kappa_1(1+\kappa_2)\Theta - \frac{3}{F(\varphi)}\alpha U(\varphi) - \frac{5F''(\varphi) + 2Z(\varphi)}{2F(\varphi)}\alpha\Pi^i\Pi_i \\ &\qquad + \frac{5F'(\varphi)}{2F(\varphi)}\alpha KK_\varphi + \frac{3F''(\varphi)}{2F(\varphi)}\alpha K^2_\varphi - \frac{3F'(\varphi)}{2F(\varphi)}A^i\Pi_i - \frac{3F'(\varphi)}{2F(\varphi)}\alpha\mathcal{L}_n K_\varphi,\end{aligned} \tag{4.176}$$

$$\begin{aligned}
&\partial_t\Theta - \beta^k\partial_k\Theta - \frac{\alpha}{2}\left[F(\varphi)R + 2D_iZ^i - F'(\varphi)D^iD_i\varphi\right] \\
&\quad = \frac{\alpha}{2}F(\varphi)\left[\frac{2}{3}K^2 - \tilde{A}^{ij}\tilde{A}^{ij}\right] - \alpha\Theta K - Z^iA_i - \alpha\kappa_1(2+\kappa_2)\Theta \\
&\quad -\frac{1}{2}Z(\varphi)\alpha K_\varphi^2 - \left[\frac{1}{2}Z(\varphi) + F''(\varphi)\right]\alpha\Pi^i\Pi_i + \alpha F'(\varphi)KK_\varphi - \alpha U(\varphi),
\end{aligned} \tag{4.177}$$

$$\begin{aligned}
&\partial_t\hat{\Gamma}^i - \beta^k\partial_k\hat{\Gamma}^i + \frac{4}{3}\alpha\tilde{\gamma}^{ij}\partial_jK - 2\alpha\tilde{\gamma}^{ki}\partial_k\Theta - \tilde{\gamma}^{kl}\partial_{(k}B^i_{l)} - \frac{1}{3}\tilde{\gamma}^{ik}\partial_{(k}B^l_{l)} + 2\alpha F'(\varphi)\tilde{\gamma}^{ij}D_jK_\varphi \\
&\quad = \frac{2}{3}\tilde{\Gamma}^iB^k_k - \tilde{\Gamma}^kB^i_k + 2\alpha F(\varphi)\left(\tilde{\Gamma}^i_{jk}\tilde{A}^{jk} - 3\tilde{A}^{ij}\frac{P_j}{\phi}\right) - 2\alpha\tilde{\gamma}^{ki}\left(\Theta\frac{A_k}{\alpha} + \frac{2}{3}KZ_k\right) \\
&\quad - 2\alpha\tilde{A}^{ij}A_j + 2\kappa_3\left(\frac{2}{3}\tilde{\gamma}^{ij}Z_jB^k_k - \tilde{\gamma}^{jk}Z_jB^i_k\right) - 2\alpha\kappa_1\tilde{\gamma}^{ij}Z_j \\
&\quad + 2\tilde{\gamma}^{ij}\left[-\alpha[Z(\varphi) + F''(\varphi)]K_\varphi\Pi_j + \alpha F'(\varphi)\tilde{A}^k_j\Pi_k + \frac{1}{3}\alpha F'(\varphi)K\Pi_j\right],
\end{aligned} \tag{4.178}$$

$$\begin{aligned}
&\partial_tK_\varphi - \beta^i\partial_iK_\varphi - \alpha\left[2Z(\varphi) + 3\frac{F'(\varphi)^2}{F(\varphi)}\right]^{-1}\left\{F'(\varphi)R - \left[2Z(\varphi) + 5\frac{F'(\varphi)^2}{F(\varphi)}\right]D_iD^i\varphi + 4\frac{F'(\varphi)}{F(\varphi)}D_iZ^i\right\} \\
&\quad = \alpha\left[2Z(\varphi) + 3\frac{F'(\varphi)^2}{F(\varphi)}\right]^{-1}\left\{F'(\varphi)[\frac{2}{3}K^2 - \tilde{A}_{ij}\tilde{A}^{ij}] + \left[Z'(\varphi) + 3\frac{F'(\varphi)F''(\varphi)}{F(\varphi)}\right]K_\varphi^2\right. \\
&\quad - \left[Z'(\varphi) + 2\frac{F'(\varphi)Z(\varphi)}{F(\varphi)} + 5\frac{F'(\varphi)F''(\varphi)}{F(\varphi)}\right]\Pi_i\Pi^i - \left[2Z(\varphi) + 3\frac{F'(\varphi)^2}{F(\varphi)}\right]\frac{A_i}{\alpha}\Pi^i \\
&\quad \left. + \left[2Z(\varphi) + 5\frac{F'(\varphi)^2}{F(\varphi)}\right]KK_\varphi - 2\frac{F'(\varphi)}{F(\varphi)}\left[3\kappa_1(1+\kappa_2)\Theta + 2\Theta K\right] + 2U'(\varphi) - 6\frac{F'(\varphi)}{F(\varphi)}U(\varphi)\right\}
\end{aligned} \tag{4.179}$$

and five equations for the auxiliary variables

$$\begin{aligned}
&\partial_tA_k - \beta^l\partial_lA_k + \alpha^2g(\alpha)\left(\partial_kK - \partial_kK_0 - 2c\partial_k\Theta\right) + \\
&\qquad\qquad = B^l_kA_l - \left[2\alpha g(\alpha) + \alpha^2g'(\alpha)\right]\left(K - K_0 - 2c\Theta\right)A_k,
\end{aligned} \tag{4.180}$$

$$\partial_tB^i_k - \beta^l\partial_lB^i_k - f\partial_kb^i = B^l_kB^i_l, \tag{4.181}$$

$$\begin{aligned}
&\partial_tD_{kij} - \beta^l\partial_lD_{kij} + s\left(-\frac{1}{2}\tilde{\gamma}_{mi}\partial_{(k}B^m_{j)} - \frac{1}{2}\tilde{\gamma}_{mj}\partial_{(k}B^m_{i)} + \frac{1}{3}\tilde{\gamma}_{ij}\partial_{(k}B^m_{m)}\right) + \alpha\partial_k\tilde{A}_{ij} \\
&\qquad\qquad = B^l_kD_{lij} + B^l_jD_{kli} + B^l_iD_{klj} - \frac{2}{3}B^l_lD_{kij} - A_k\left(\tilde{A}_{ij} - \frac{1}{3}\tilde{\gamma}_{ij}\,\mathrm{tr}\,\tilde{A}\right),
\end{aligned} \tag{4.182}$$

$$\partial_tP_k - \beta^l\partial_lP_k - \frac{1}{3}\phi\left(\alpha\partial_kK - \partial_{(k}B^l_{l)}\right) = B^l_kP_l + \frac{1}{3}\left(\alpha K - B^l_l\right)P_k + \frac{1}{3}\phi KA_k, \tag{4.183}$$

$$\partial_t\Pi_i - \beta^k\partial_k\Pi_i + \alpha\partial_iK_\varphi = -A_iK_\varphi + B^k_i\Pi_k. \tag{4.184}$$

In the equations above we use blue colour for the new terms arising from the scalar field, except in the equation of K_φ which is completely new. Additionally, we use red colour to indicate terms added to enforce the algebraic constraints ($\det(\tilde{\gamma}_{ij}) = 1$ and $\mathrm{tr}\tilde{A}_{ij} = 0$).

The hyperbolicity of the system needs to be re-examined due to the existence of the scalar field and that is why we remove those extra terms in red colour in equation (4.71)-(4.83). Those terms are added solely for symmetry enhancement, and they may take completely different forms in the scalar field theory. Therefore it makes little sense to keep them in the evolving system for now. The construction of the initial condition is not covered here for a similar reason. The constraints we need to solve for the initial condition can be derived from (4.159) and (4.160) in a straightforward way, but the Bowen–York approach we introduced in section 4.1.2 is currently only applicable to the standard gravity. it is very likely that the momentum constraint does not admit an analytical solution anymore and a numerical solution of the momentum would be necessary. Also, the puncture method may lose its good properties given the different shapes of the Hamiltonian constraint, and the gauge condition requires a reassessment as well. There are still many challenges in simulating black holes under scalar field gravity. We plan to explore this topic deeper in the future, once our application is fully functional within the scope of the standard gravity.

CHAPTER 5

Summary and Outlook

5.1 Summary

In this thesis, we present the two astrophysical applications implemented on the platform ExaHyPE2.

- In our spherical collapse application, we have studied the spherical accretion of collisional gas in both an Einstein de-Sitter universe and SST gravity model. Notably, we have successfully derived self-similar solutions for some special cases within the latter class of models. This is the first time such solutions are found and reported as far as our knowledge goes. The simulation output also yields good agreements with the theoretical prediction we found. This thus not only verifies the existence of self-similarity in the considered models but also proves the reliability and correctness of our application.
- In our numerical relativity application, we have looked into various simulation cases, including the gauge wave, the single Schwarzschild black hole and the head-on collision of binary black holes. Additionally, we performed a binary black hole merger and reported its preliminary results, as well as its prediction

of the gravitational wave. With the code development we covered in the corresponding chapter, our application has already shown the capability to carry out simulations of vacuum and black hole spacetimes. However, certain issues affecting the long-term stability of static systems still exist, and efforts to fully resolve these issues are ongoing.

In addition to reporting our astrophysical application, we also presented the technical development of the ExaHyPE2 engine itself during the implementation of these applications. We improved the old finite volume solver and wrote a new solver based on the fourth-order finite difference scheme. We also incorporated new Sommerfeld-type boundary conditions and a refinement transition strategy compatible with the multiple-layer halo. These enhancements successfully suppressed the numerical fluctuations and reflections at the domain boundary and the AMR boundary. Furthermore, we also deployed and tested the tracer module in ExaHyPE2, which provides us with better flexibility on the data output of our simulations. Those technical advancements are compatible with other applications based on the engine, and enhance the overall capability of ExaHyPE2 to address future problems in a wide range of scientific and engineering domains.

5.2 Outlook

We are still actively developing our applications, aiming to solve the existing issues and improve their efficiency. Furthermore, we plan to incorporate more realistic and complex astrophysical scenarios in our applications as well.

- In the spherical collapse application, we now have a working hydrodynamical simulation code, where new models of gravity can be straightforwardly implemented. A natural next step based on that is to run simulations for more realistic modified gravity models that do not have self-similar solutions, including the original DGP model, the K-mouflage model and the chameleon model. For the latter, we may need to either add a multigrid solver for the scalar field or adopt some approximate solutions such as the thin-shell solution. In a future project, we will compare the collapse of collisional gas in these different models in detail. If the above speculation, namely the spherical solutions rescaled by the true turnaround radii of models are approximately the same in different cosmological models, turns out to be correct, then the differences in the physical solutions of these models can be largely ascribed to the differences in their turnaround radii, which might offer a simple way to model the modified gravity effects. In addition, we plan to add more physical processes, such as radiative cooling (e.g., Abadi, Bower, and Navarro, 2000), in the code, to understand how they interfere with the effects of a modified law of gravity. Altogether, these will hopefully offer us new insights into the behaviour of gas, and hence the galaxy formation process, in modified gravity models.

- In the numerical relativity application, addressing the instability issues that affect long-term performances is still our primary focus for now. Once this issue is successfully resolved, we shall achieve a stable and properly running code for black hole simulations in standard gravity and be able to extract gravitational wave signals from those systems. Our following plan is to include modified gravity theories in these systems. While this thesis has reported some of the preliminary explorations of the scalar-tensor theory of gravity (section 4.5), there is still substantial work that needs to be done for a numerical relativity code that can perform simulations of binary black hole mergers in this framework. For instance, the hyperbolicity of the evolution system and the choices of the gauge condition need to be reassessed; the current approach to constructing the initial conditions using the puncture method may no longer be valid and the alternative technique such as black hole excision may need to be implemented in future simulations. These considerations in physics require further development of our numerical scheme and code. Moreover, we also have plans to incorporate the matter components in our application, expanding our simulation scenarios to include other compact objects in the universe, such as neutron stars, and the accretion around them. By further combining these advancements with the implementation of modified gravity, our application will have the capability to examine gravity theories in the strong-field regimes and provide prediction and insights into future gravitational wave detection projects involving compact objects.

Besides the advancements of the astrophysical applications, we also have plans to further enhance the technical capacity of ExaHyPE2 engine. There are several new features that are in active development. For example, ExaHyPE2 will soon provide more advanced computation solvers, such as local time-stepping based on resolution or eigenvalues, and the arbitrary-high-order-method-using-derivatives (ADER) DG method. These solvers will provide higher accuracy and efficiency in future simulations. The solver coupling framework in ExaHyPE2 is also under test, which will enable users to implement multiple solvers simultaneously in their simulations. The different solvers can help correct the evolutions of each other, resulting in better simulation quality. Additionally, users will also have the ability to manipulate the data transfer between patches more closely during timesteps, which provides a potential approach to implement higher order PDE formulation on ExaHyPE2.

Bibliography

[1] Mario G. Abadi, Richard G. Bower, and Julio F. Navarro. “Self-similar shocked accretion of collisional gas with radiative cooling”. In: MNRAS 314.4 (June 2000), pp. 759–767. DOI: 10.1046/j.1365-8711.2000.03349.x. arXiv: astro-ph/9904364 [astro-ph].

[2] C. Alard. “Equilibrium solution for cold dynamical systems and self-similarity”. In: *arXiv e-prints*, arXiv:2010.09400 (Oct. 2020), arXiv:2010.09400. arXiv: 2010.09400 [nlin.PS].

[3] Miguel Alcubierre. “276GRAVITATIONAL WAVE EXTRACTION”. In: *Introduction to 3+1 Numerical Relativity*. Oxford University Press, Apr. 2008. ISBN: 9780199205677. DOI: 10.1093/acprof:oso/9780199205677.003.0008. eprint: https://academic.oup.com/book/0/chapter/156750056/chapter-ag-pdf/44946957/book_9640_section_156750056.ag.pdf. URL: https://doi.org/10.1093/acprof:oso/9780199205677.003.0008.

[4] Miguel Alcubierre and Bernd Brügmann. “Simple excision of a black hole in $3 + 1$ numerical relativity”. In: *Phys. Rev. D* 63 (10 2001), p. 104006. DOI: 10.1103/PhysRevD.63.104006. URL: https://link.aps.org/doi/10.1103/PhysRevD.63.104006.

[5] Miguel Alcubierre et al. “Dynamical evolution of quasicircular binary black hole data”. In: *Phys. Rev. D* 72 (4 2005), p. 044004. DOI: 10.1103/PhysRevD.

72.044004. URL: https://link.aps.org/doi/10.1103/PhysRevD.72.044004.

[6] Miguel Alcubierre et al. “Gauge conditions for long-term numerical black hole evolutions without excision”. In: *Phys. Rev. D* 67 (8 2003), p. 084023. DOI: 10.1103/PhysRevD.67.084023. URL: https://link.aps.org/doi/10.1103/PhysRevD.67.084023.

[7] Miguel Alcubierre et al. “Towards standard testbeds for numerical relativity”. In: *Classical and Quantum Gravity* 21.2 (2003), p. 589. DOI: 10.1088/0264-9381/21/2/019. URL: https://dx.doi.org/10.1088/0264-9381/21/2/019.

[8] Daniela Alic et al. “Conformal and covariant formulation of the Z4 system with constraint-violation damping”. In: *Phys. Rev. D* 85 (6 2012), p. 064040. DOI: 10.1103/PhysRevD.85.064040. URL: https://link.aps.org/doi/10.1103/PhysRevD.85.064040.

[9] Luca Amendola and Shinji Tsujikawa. *Dark energy: Theory and observations*. English. Publisher Copyright: © L. Amendola and S. Tsujikawa 2010. Cambridge University Press, Jan. 2010. ISBN: 9780521516006. DOI: 10.1017/CBO9780511750823.

[10] Marcus Ansorg, Bernd Brügmann, and Wolfgang Tichy. “Single-domain spectral method for black hole puncture data”. In: *Phys. Rev. D* 70 (6 2004), p. 064011. DOI: 10.1103/PhysRevD.70.064011. URL: https://link.aps.org/doi/10.1103/PhysRevD.70.064011.

[11] V. Antonuccio-Delogu and S. Colafrancesco. “Dynamical Friction, Secondary Infall, and the Evolution of Clusters of Galaxies”. In: ApJ 427 (May 1994), p. 72. DOI: 10.1086/174122.

[12] Richard Arnowitt, Stanley Deser, and Charles W. Misner. “Republication of: The dynamics of general relativity”. In: *General Relativity and Gravitation* 40.9 (Sept. 2008), pp. 1997–2027. DOI: 10.1007/s10714-008-0661-1. arXiv: gr-qc/0405109 [gr-qc].

[13] Gerard Auger and Eric Plagnol. *Overview Of Gravitational Waves, An: Theory, Sources And Detection*. World Scientific, 2017.

[14] E. Babichev, C. Deffayet, and R. Ziour. “k-Mouflage gravity”. In: *IJMPD* 18 (2009), pp. 2147–2154. DOI: `10.1142/S0218271809016107`. arXiv: `0905.2943 [hep-th]`.

[15] John Baker, Manuela Campanelli, and Carlos Lousto. “The Lazarus project: A pragmatic approach to binary black hole evolutions”. In: *Physical Review D* 65 (May 2001). DOI: `10.1103/PhysRevD.65.044001`.

[16] John G. Baker et al. “Gravitational-Wave Extraction from an Inspiraling Configuration of Merging Black Holes”. In: *Phys. Rev. Lett.* 96 (11 2006), p. 111102. DOI: `10.1103/PhysRevLett.96.111102`. URL: `https://link.aps.org/doi/10.1103/PhysRevLett.96.111102`.

[17] Alexandre Barreira et al. “Halo model and halo properties in Galileon gravity cosmologies”. In: *JCAP* 04 (2014), p. 029. DOI: `10.1088/1475-7516/2014/04/029`. arXiv: `1401.1497 [astro-ph.CO]`.

[18] Thomas W. Baumgarte and Stuart L. Shapiro. “Numerical integration of Einstein’s field equations”. In: *Phys. Rev. D* 59 (2 1998), p. 024007. DOI: `10.1103/PhysRevD.59.024007`. URL: `https://link.aps.org/doi/10.1103/PhysRevD.59.024007`.

[19] Thomas W. Baumgarte and Stuart L. Shapiro. “Numerical integration of Einstein’s field equations”. In: *Phys. Rev. D* 59 (2 1998), p. 024007. DOI: `10.1103/PhysRevD.59.024007`. URL: `https://link.aps.org/doi/10.1103/PhysRevD.59.024007`.

[20] T.W. Baumgarte and S.L. Shapiro. *Numerical Relativity: Solving Einstein’s Equations on the Computer*. Cambridge University Press, 2010. ISBN: 9780521514071. URL: `https://books.google.co.uk/books?id=dxU1OEinvRUC`.

[21] E. Bertschinger. “Self-similar secondary infall and accretion in an Einstein-de Sitter universe”. In: *Astrophys. J., Suppl. Ser.; (United States)* 58 (May 1985), pp. 39–65. DOI: `10.1086/191028`.

[22] Peter Bodenheimer et al. *Numerical methods in astrophysics: an introduction.* CRC Press, 2006.

[23] C. Bona et al. "General-covariant evolution formalism for numerical relativity". In: *Phys. Rev. D* 67 (10 2003), p. 104005. DOI: 10.1103/PhysRevD.67.104005. URL: https://link.aps.org/doi/10.1103/PhysRevD.67.104005.

[24] C. Bona et al. "General-covariant evolution formalism for numerical relativity". In: *Phys. Rev. D* 67 (10 2003), p. 104005. DOI: 10.1103/PhysRevD.67.104005. URL: https://link.aps.org/doi/10.1103/PhysRevD.67.104005.

[25] Carles Bona et al. "New Formalism for Numerical Relativity". In: *Phys. Rev. Lett.* 75 (4 1995), pp. 600–603. DOI: 10.1103/PhysRevLett.75.600. URL: https://link.aps.org/doi/10.1103/PhysRevLett.75.600.

[26] Silvano Bonazzola et al. "Constrained scheme for the Einstein equations based on the Dirac gauge and spherical coordinates". In: *Phys. Rev. D* 70 (10 2004), p. 104007. DOI: 10.1103/PhysRevD.70.104007. URL: https://link.aps.org/doi/10.1103/PhysRevD.70.104007.

[27] Stefano Borgani and Andrey Kravtsov. "Cosmological Simulations of Galaxy Clusters". In: *Advanced Science Letters* 4.2 (Feb. 2011), pp. 204–227. DOI: 10.1166/asl.2011.1209. arXiv: 0906.4370 [astro-ph.CO].

[28] Jeffrey M. Bowen and James W. York. "Time-asymmetric initial data for black holes and black-hole collisions". In: *Phys. Rev. D* 21 (8 1980), pp. 2047–2056. DOI: 10.1103/PhysRevD.21.2047. URL: https://link.aps.org/doi/10.1103/PhysRevD.21.2047.

[29] Michael Boyle et al. "High-accuracy comparison of numerical relativity simulations with post-Newtonian expansions". In: *Phys. Rev. D* 76 (12 2007), p. 124038. DOI: 10.1103/PhysRevD.76.124038. URL: https://link.aps.org/doi/10.1103/PhysRevD.76.124038.

[30] Steven Brandt and Bernd Brügmann. "A Simple Construction of Initial Data for Multiple Black Holes". In: *Phys. Rev. Lett.* 78 (19 1997), pp. 3606–3609. DOI: 10.1103/PhysRevLett.78.3606. URL: https://link.aps.org/doi/10.1103/PhysRevLett.78.3606.

[31] Johann S. Brauchart and Peter J. Grabner. "Distributing many points on spheres: Minimal energy and designs". In: *Journal of Complexity* 31.3 (2015), pp. 293–326. ISSN: 0885064X. DOI: 10.1016/j.jco.2015.02.003. URL: http://www.sciencedirect.com/science/article/pii/S0885064X15000205.

[32] Philippe Brax. "Screening mechanisms in modified gravity". In: *Classical and Quantum Gravity* 30.21, 214005 (Nov. 2013), p. 214005. DOI: 10.1088/0264-9381/30/21/214005.

[33] Philippe Brax, Clare Burrage, and Anne-Christine Davis. "Screening fifth forces in k-essence and DBI models". In: *JCAP* 01 (2013), p. 020. DOI: 10.1088/1475-7516/2013/01/020. arXiv: 1209.1293 [hep-th].

[34] Dieter R. Brill and Richard W. Lindquist. "Interaction Energy in Geometrostatics". In: *Phys. Rev.* 131 (1 1963), pp. 471–476. DOI: 10.1103/PhysRev.131.471. URL: https://link.aps.org/doi/10.1103/PhysRev.131.471.

[35] Bernd Brügmann et al. "Calibration of moving puncture simulations". In: *Phys. Rev. D* 77 (2 2008), p. 024027. DOI: 10.1103/PhysRevD.77.024027. URL: https://link.aps.org/doi/10.1103/PhysRevD.77.024027.

[36] Hans-Joachim Bungartz, Miriam Mehl, and Tobias Weinzierl. "A Parallel Adaptive Cartesian PDE Solver Using Space–Filling Curves". In: *Euro-Par 2006 Parallel Processing*. Ed. by Wolfgang E. Nagel, Wolfgang V. Walter, and Wolfgang Lehner. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1064–1074. ISBN: 978-3-540-37784-9.

[37] M. Campanelli, C. O. Lousto, and Y. Zlochower. "Last orbit of binary black holes". In: *Phys. Rev. D* 73 (6 2006), p. 061501. DOI: 10.1103/PhysRevD.73.061501. URL: https://link.aps.org/doi/10.1103/PhysRevD.73.061501.

[38] M. Campanelli et al. "Accurate Evolutions of Orbiting Black-Hole Binaries without Excision". In: *Phys. Rev. Lett.* 96 (11 2006), p. 111101. DOI: 10.1103/PhysRevLett.96.111101. URL: https://link.aps.org/doi/10.1103/PhysRevLett.96.111101.

[39] D.E. Charrier, B. Hazelwood, and T. Weinzierl. “Enclave Tasking for DG Methods on Dynamically Adaptive Meshes”. In: *SIAM Journal on Scientific Computing* 42.3 (2020), pp. C69–C96.

[40] Demetrios Christodoulou. “Reversible and Irreversible Transformations in Black-Hole Physics”. In: *Phys. Rev. Lett.* 25 (22 1970), pp. 1596–1597. DOI: `10.1103/PhysRevLett.25.1596`. URL: `https://link.aps.org/doi/10.1103/PhysRevLett.25.1596`.

[41] Katy Clough et al. “GRChombo: Numerical relativity with adaptive mesh refinement”. In: *Classical and Quantum Gravity* 32.24 (2015), p. 245011. DOI: `10.1088/0264-9381/32/24/245011`. URL: `https://doi.org/10.1088/0264-9381/32/24/245011`.

[42] Omar Contigiani, Valeri Vardanyan, and Alessandra Silvestri. “Splashback radius in symmetron gravity”. In: 99.6, 064030 (Mar. 2019), p. 064030. DOI: `10.1103/PhysRevD.99.064030`. arXiv: `1812.05568 [astro-ph.CO]`.

[43] A. Dedner et al. “Hyperbolic Divergence Cleaning for the MHD Equations”. In: *Journal of Computational Physics* 175.2 (2002), pp. 645 –673. ISSN: 0021-9991. DOI: `https://doi.org/10.1006/jcph.2001.6961`. URL: `http://www.sciencedirect.com/science/article/pii/S002199910196961X`.

[44] C. Deffayet, Gilles Esposito-Farese, and A. Vikman. “Covariant Galileon”. In: *Phys. Rev. D* 79 (2009), p. 084003. DOI: `10.1103/PhysRevD.79.084003`. arXiv: `0901.1314 [hep-th]`.

[45] A. Del Popolo, F. Pace, and J. A. S. Lima. “Spherical collapse model with shear and angular momentum in dark energy cosmologies”. In: *Monthly Notices of the Royal Astronomical Society* 430.1 (Jan. 2013), pp. 628–637. ISSN: 0035-8711. DOI: `10.1093/mnras/sts669`. eprint: `https://academic.oup.com/mnras/article-pdf/430/1/628/3094989/sts669.pdf`. URL: `https://doi.org/10.1093/mnras/sts669`.

[46] A. Dubey et al. “A Survey of High Level Frameworks in Block-Structured Adaptive Mesh Refinement Packages”. In: *CoRR* 74.12 (2016), pp. 3217–3227.

[47] Matthew D. Duez et al. “Hydrodynamic simulations in $3+1$ general relativity”. In: *Phys. Rev. D* 67 (2 2003), p. 024004. DOI: 10.1103/PhysRevD.67.024004. URL: https://link.aps.org/doi/10.1103/PhysRevD.67.024004.

[48] Michael Dumbser et al. “Conformal and covariant Z4 formulation of the Einstein equations: Strongly hyperbolic first-order reduction and solution with discontinuous Galerkin schemes”. In: *Phys. Rev. D* 97 (8 2018), p. 084053. DOI: 10.1103/PhysRevD.97.084053. URL: https://link.aps.org/doi/10.1103/PhysRevD.97.084053.

[49] Gia Dvali, Gregory Gabadadze, and Massimo Porrati. “4D gravity on a brane in 5D Minkowski space”. In: *Physics Letters B* 485.1 (2000), pp. 208–214. ISSN: 0370-2693. DOI: https://doi.org/10.1016/S0370-2693(00)00669-9. URL: https://www.sciencedirect.com/science/article/pii/S0370269300006699.

[50] J. A. Fillmore and P. Goldreich. “Self-similar gravitational collapse in an expanding universe”. In: *Astrophysical Journal* 281 (June 1984), pp. 1–8. DOI: 10.1086/162070.

[51] V. Gayathri et al. “Eccentricity estimate for black hole mergers with numerical relativity simulations”. In: *Nature Astronomy* 6.3 (2022), pp. 344–349. ISSN: 2397-3366. DOI: 10.1038/s41550-021-01568-w. URL: https://doi.org/10.1038/s41550-021-01568-w.

[52] James E. Gunn and III Gott J. Richard. “On the Infall of Matter Into Clusters of Galaxies and Some Effects on Their Evolution”. In: ApJ 176 (Aug. 1972), p. 1. DOI: 10.1086/151605.

[53] A. Halle, S. Colombi, and S. Peirani. “Phase-space structure analysis of self-gravitating collisionless spherical systems”. In: *Astronomy & Astrophysics* 621, A8 (Jan. 2019), A8. DOI: 10.1051/0004-6361/201833460. arXiv: 1701.01384 [astro-ph.GA].

[54] Mark Hannam et al. “Geometry and Regularity of Moving Punctures”. In: *Phys. Rev. Lett.* 99 (24 2007), p. 241102. DOI: 10.1103/PhysRevLett.99.

241102. URL: https://link.aps.org/doi/10.1103/PhysRevLett.99.241102.

[55] James Healy and Carlos O. Lousto. "Remnant of binary black-hole mergers: New simulations and peak luminosity studies". In: *Phys. Rev. D* 95 (2 2017), p. 024037. DOI: 10.1103/PhysRevD.95.024037. URL: https://link.aps.org/doi/10.1103/PhysRevD.95.024037.

[56] Lavinia Heisenberg. "Generalization of the Proca Action". In: *J. Cosmo. Astropart. Phys.* 1405 (2014), p. 015. DOI: 10.1088/1475-7516/2014/05/015. arXiv: 1402.7026 [hep-th].

[57] David Hilditch et al. "Compact binary evolutions with the Z4c formulation". In: *Phys. Rev. D* 88 (8 2013), p. 084057. DOI: 10.1103/PhysRevD.88.084057. URL: https://link.aps.org/doi/10.1103/PhysRevD.88.084057.

[58] Justin Khoury and Amanda Weltman. "Chameleon cosmology". In: *Phys. Rev. D* 69 (2004), p. 044026. DOI: 10.1103/PhysRevD.69.044026. arXiv: astro-ph/0309411.

[59] Justin Khoury and Amanda Weltman. "Chameleon fields: Awaiting surprises for tests of gravity in space". In: *Phys. Rev. Lett.* 93 (2004), p. 171104. DOI: 10.1103/PhysRevLett.93.171104. arXiv: astro-ph/0309300.

[60] Kazuya Koyama. "TOPICAL REVIEW: Ghosts in the self-accelerating universe". In: *Classical and Quantum Gravity* 24.24 (Dec. 2007), R231–R253. DOI: 10.1088/0264-9381/24/24/R01. arXiv: 0709.2399 [hep-th].

[61] Kazuya Koyama and Fabio P Silva. "Nonlinear interactions in a cosmological background in the Dvali-Gabadadze-Porrati braneworld". In: *Phys. Rev. D* 75 (8 2007), p. 084040. DOI: 10.1103/PhysRevD.75.084040. URL: https://link.aps.org/doi/10.1103/PhysRevD.75.084040.

[62] H. Kreiss and J. Oliger. *Methods for the approximate solution of time dependent problems*. GARP Publications Series. SELBSTVERL.) FEBR, 1973. URL: https://books.google.co.uk/books?id=wj9szwEACAAJ.

[63] M. Le Delliou and R. N. Henriksen. "Non-radial motion and the NFW profile". In: A&A 408 (Sept. 2003), pp. 27–38. DOI: 10.1051/0004-6361:20030922. arXiv: astro-ph/0307046 [astro-ph].

[64] Randall J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge Texts in Applied Mathematics. Cambridge University Press, 2002. DOI: 10.1017/CBO9780511791253.

[65] Baojiu Li and George Efstathiou. "An extended excursion set approach to structure formation in chameleon models". In: MNRAS 421.2 (Apr. 2012), pp. 1431–1442. DOI: 10.1111/j.1365-2966.2011.20404.x. arXiv: 1110.6440 [astro-ph.CO].

[66] Baojiu Li, Gong-Bo Zhao, and Kazuya Koyama. "Exploring Vainshtein mechanism on adaptively refined meshes". In: J. Cosmology Astropart. Phys. 2013.5, 023 (May 2013), p. 023. DOI: 10.1088/1475-7516/2013/05/023. arXiv: 1303.0008 [astro-ph.CO].

[67] Baojiu Li et al. "Dynamic task fusion for a block-structured finite volume solver over a dynamically adaptive mesh with local time stepping". In: *ISC High Performance 2022* (2022). (accepted).

[68] Lee Lindblom et al. "A new generalized harmonic evolution system". In: *Classical and Quantum Gravity* 23.16 (2006), S447–S462. DOI: 10.1088/0264-9381/23/16/s09. URL: https://doi.org/10.1088%2F0264-9381%2F23%2F16%2Fs09.

[69] Eric V. Linder. "Einstein's other gravity and the acceleration of the Universe". In: *Phys. Rev. D* 81 (12 2010), p. 127301. DOI: 10.1103/PhysRevD.81.127301. URL: https://link.aps.org/doi/10.1103/PhysRevD.81.127301.

[70] Lucas Lombriser, Kazuya Koyama, and Baojiu Li. "Halo modelling in chameleon theories". In: *JCAP* 03 (2014), p. 021. DOI: 10.1088/1475-7516/2014/03/021. arXiv: 1312.1292 [astro-ph.CO].

[71] Rafael C.C. Lopes et al. “Turnaround radius in f(R) model”. In: *Journal of Cosmology and Astroparticle Physics* 2018.09 (2018), pp. 010–010. DOI: `10.1088/1475-7516/2018/09/010`. URL: `https://doi.org/10.1088/1475-7516/2018/09/010`.

[72] Carlos O. Lousto, James Healy, and Hiroyuki Nakano. “Spin flips in generic black hole binaries”. In: *Phys. Rev. D* 93 (4 2016), p. 044031. DOI: `10.1103/PhysRevD.93.044031`. URL: `https://link.aps.org/doi/10.1103/PhysRevD.93.044031`.

[73] Carlos O. Lousto and Yosef Zlochower. “Black hole binary remnant mass and spin: A new phenomenological formula”. In: *Phys. Rev. D* 89 (10 2014), p. 104052. DOI: `10.1103/PhysRevD.89.104052`. URL: `https://link.aps.org/doi/10.1103/PhysRevD.89.104052`.

[74] Frank Löffler et al. “The Einstein Toolkit: a community computational infrastructure for relativistic astrophysics”. In: *Classical and Quantum Gravity* 29.11 (2012), p. 115001. DOI: `10.1088/0264-9381/29/11/115001`.

[75] Hugo Martel and Paul R. Shapiro. “A convenient set of comoving cosmological variables and their application”. In: MNRAS 297.2 (June 1998), pp. 467–485. DOI: `10.1046/j.1365-8711.1998.01497.x`. arXiv: `astro-ph/9710119 [astro-ph]`.

[76] Matthew C. Martino, Hans F. Stabenau, and Ravi K. Sheth. “Spherical collapse and cluster counts in modified gravity models”. In: 79.8, 084013 (Apr. 2009), p. 084013. DOI: `10.1103/PhysRevD.79.084013`. arXiv: `0812.0200 [astro-ph]`.

[77] I. G. McCarthy et al. “Modelling shock heating in cluster mergers - I. Moving beyond the spherical accretion model”. In: MNRAS 376.2 (Apr. 2007), pp. 497–522. DOI: `10.1111/j.1365-2966.2007.11465.x`. arXiv: `astro-ph/0701335 [astro-ph]`.

[78] Ian G. McCarthy et al. “The BAHAMAS project: Calibrated hydrodynamical simulations for large-scale structure cosmology”. In: *Mon. Not. Roy. Astron.*

Soc. 465.3 (2017), pp. 2936–2965. DOI: 10.1093/mnras/stw2792. arXiv: 1603.02702 [astro-ph.CO].

[79] Peter McCorquodale and Phillip Colella. "A high-order finite-volume method for conservation laws on locally refined grids". In: *Communications in Applied Mathematics and Computational Science* 6.1 (2011), pp. 1–25.

[80] Abdul H. Mroué et al. "Measuring orbital eccentricity and periastron advance in quasicircular black hole simulations". In: *Phys. Rev. D* 82 (12 2010), p. 124016. DOI: 10.1103/PhysRevD.82.124016. URL: https://link.aps.org/doi/10.1103/PhysRevD.82.124016.

[81] Takashi Nakamura, Kenichi Oohara, and Yasufumi Kojima. "General Relativistic Collapse to Black Holes and Gravitational Waves from Black Holes". In: *Progress of Theoretical Physics Supplement* 90 (Jan. 1987), pp. 1–218. ISSN: 0375-9687. DOI: 10.1143/PTPS.90.1.

[82] Ezra Newman and Roger Penrose. "An Approach to Gravitational Radiation by a Method of Spin Coefficients". In: *Journal of Mathematical Physics* 3.3 (Dec. 2004), pp. 566–578. ISSN: 0022-2488. DOI: 10.1063/1.1724257. eprint: https://pubs.aip.org/aip/jmp/article-pdf/3/3/566/11121878/566_1_online.pdf. URL: https://doi.org/10.1063/1.1724257.

[83] Alberto Nicolis, Riccardo Rattazzi, and Enrico Trincherini. "The Galileon as a local modification of gravity ". In: *Phys. Rev.* D79 (2009), p. 064036. DOI: 10.1103/PhysRevD.79.064036. arXiv: 0811.2197 [hep-th].

[84] Maria Okounkova et al. "Gravitational wave inference on a numerical-relativity simulation of a black hole merger beyond general relativity". In: *Phys. Rev. D* 107 (2 2023), p. 024046. DOI: 10.1103/PhysRevD.107.024046. URL: https://link.aps.org/doi/10.1103/PhysRevD.107.024046.

[85] Francesco Pace, Ronaldo C. Batista, and Antonino Del Popolo. "Effects of shear and rotation on the spherical collapse model for clustering dark energy". In: *Monthly Notices of the Royal Astronomical Society* 445.1 (Sept. 2014), pp. 648–659. ISSN: 0035-8711. DOI: 10.1093/mnras/stu1782. eprint:

https://academic.oup.com/mnras/article-pdf/445/1/648/18469019/stu1782.pdf. URL: https://doi.org/10.1093/mnras/stu1782.

[86] A. Del Popolo. "THE CUSP/CORE PROBLEM AND THE SECONDARY INFALL MODEL". In: *The Astrophysical Journal* 698.2 (2009), pp. 2093–2113. DOI: 10.1088/0004-637x/698/2/2093. URL: https://doi.org/10.1088/0004-637x/698/2/2093.

[87] W. H. Press et al. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. 3rd. New York, NY, USA: Cambridge University Press, 2007.

[88] Frans Pretorius. "Evolution of Binary Black-Hole Spacetimes". In: *Phys. Rev. Lett.* 95 (12 2005), p. 121101. DOI: 10.1103/PhysRevLett.95.121101. URL: https://link.aps.org/doi/10.1103/PhysRevLett.95.121101.

[89] Miren Radia et al. "Lessons for adaptive mesh refinement in numerical relativity". In: *Classical and Quantum Gravity* 39.13 (2022), p. 135006. DOI: 10.1088/1361-6382/ac6fa9. URL: https://doi.org/10.1088/1361-6382/ac6fa9.

[90] Anne Reinarz et al. "ExaHyPE: An engine for parallel dynamically adaptive simulations of wave problems". In: *Computer Physics Communications* 254 (2020), p. 107251. ISSN: 0010-4655. DOI: https://doi.org/10.1016/j.cpc.2020.107251. URL: https://www.sciencedirect.com/science/article/pii/S001046552030076X.

[91] Luciano Rezzolla et al. "Final spin from the coalescence of two black holes". In: *Phys. Rev. D* 78 (4 2008), p. 044002. DOI: 10.1103/PhysRevD.78.044002. URL: https://link.aps.org/doi/10.1103/PhysRevD.78.044002.

[92] Barbara S. Ryden. "Galaxy Formation: The Role of Tidal Torques and Dissipational Infall". In: *Astrophysical Journal* 329 (June 1988), p. 589. DOI: 10.1086/166406.

[93] Barbara S. Ryden and James E. Gunn. "Galaxy Formation by Gravitational Collapse". In: *Astrophysical Journal* 318 (July 1987), p. 15. DOI: 10.1086/165349.

[94] Joop Schaye et al. “The EAGLE project: Simulating the evolution and assembly of galaxies and their environments”. In: *Mon. Not. Roy. Astron. Soc.* 446 (2015), pp. 521–554. DOI: 10.1093/mnras/stu2058. arXiv: 1407.7040 [astro-ph.GA].

[95] Mark A. Scheel et al. “Solving Einstein’s equations with dual coordinate frames”. In: *Phys. Rev. D* 74 (10 2006), p. 104006. DOI: 10.1103/PhysRevD.74.104006. URL: https://link.aps.org/doi/10.1103/PhysRevD.74.104006.

[96] Fabian Schmidt, Wayne Hu, and Marcos Lima. “Spherical collapse and the halo model in braneworld gravity”. In: 81.6, 063005 (Mar. 2010), p. 063005. DOI: 10.1103/PhysRevD.81.063005. arXiv: 0911.5178 [astro-ph.CO].

[97] Holger Schulz et al. “Task inefficiency patterns for a wave equation solver”. In: *IWOMP* (2021).

[98] Masaru Shibata. *Numerical Relativity.* Vol. 1. World Scientific, 2015.

[99] Masaru Shibata and Takashi Nakamura. “Evolution of three-dimensional gravitational waves: Harmonic slicing case”. In: *Phys. Rev. D* 52 (10 1995), pp. 5428–5444. DOI: 10.1103/PhysRevD.52.5428. URL: https://link.aps.org/doi/10.1103/PhysRevD.52.5428.

[100] Masaru Shibata and Takashi Nakamura. “Evolution of three-dimensional gravitational waves: Harmonic slicing case”. In: *Phys. Rev. D* 52 (10 1995), pp. 5428–5444. DOI: 10.1103/PhysRevD.52.5428. URL: https://link.aps.org/doi/10.1103/PhysRevD.52.5428.

[101] P. Sikivie, I. I. Tkachev, and Yun Wang. “Secondary infall model of galactic halo formation and the spectrum of cold dark matter particles on Earth”. In: *Phys. Rev. D* 56 (4 1997), pp. 1863–1878. DOI: 10.1103/PhysRevD.56.1863. URL: https://link.aps.org/doi/10.1103/PhysRevD.56.1863.

[102] Larry Smarr. “SPACE-TIMES GENERATED BY COMPUTERS: BLACK HOLES WITH GRAVITATIONAL RADIATION*”. In: *Annals of the New York Academy of Sciences* 302.1 (1977), pp. 569–604. DOI: 10.1111/j.1749-6632.1977.tb37076.x.

[103] Larry Smarr and James W. York. "Radiation gauge in general relativity". In: *Phys. Rev. D* 17 (8 1978), pp. 1945–1956. DOI: `10.1103/PhysRevD.17.1945`. URL: `https://link.aps.org/doi/10.1103/PhysRevD.17.1945`.

[104] Yong-Seon Song, Ignacy Sawicki, and Wayne Hu. "Large-scale tests of the Dvali-Gabadadze-Porrati model". In: 75.6, 064003 (Mar. 2007), p. 064003. DOI: `10.1103/PhysRevD.75.064003`. arXiv: `astro-ph/0606286 [astro-ph]`.

[105] Thomas P. Sotiriou and Valerio Faraoni. "$f(R)$ theories of gravity". In: *Rev. Mod. Phys.* 82 (1 2010), pp. 451–497. DOI: `10.1103/RevModPhys.82.451`. URL: `https://link.aps.org/doi/10.1103/RevModPhys.82.451`.

[106] Volker Springel et al. "First results from the IllustrisTNG simulations: matter and galaxy clustering". In: *Mon. Not. Roy. Astron. Soc.* 475.1 (2018), pp. 676–698. DOI: `10.1093/mnras/stx3304`. arXiv: `1707.03397 [astro-ph.GA]`.

[107] Kandaswamy Subramanian, Renyue Cen, and Jeremiah P. Ostriker. "The Structure of Dark Matter Halos in Hierarchical Clustering Theories". In: *The Astrophysical Journal* 538.2 (2000), pp. 528–542. DOI: `10.1086/309152`. URL: `https://doi.org/10.1086/309152`.

[108] H. Sundar, R. S. Sampath, and G. Biros. "Bottom-Up Construction and 2:1 Balance Refinement of Linear Octrees in Parallel". In: *SIAM Journal on Scientific Computing* 30.5 (2008), pp. 2675–2708.

[109] Béla Szilágyi et al. "An explicit harmonic code for black-hole evolution using excision". In: *Classical and Quantum Gravity* 24.12 (2007), S275–S293. DOI: `10.1088/0264-9381/24/12/s18`. URL: `https://doi.org/10.1088%2F0264-9381%2F24%2F12%2Fs18`.

[110] R. Teyssier. "Cosmological hydrodynamics with adaptive mesh refinement. A new high resolution code called RAMSES". In: *Astronomy & Astrophysics* 385 (Apr. 2002), pp. 337–364. DOI: `10.1051/0004-6361:20011817`. arXiv: `astro-ph/0111367 [astro-ph]`.

[111] Wolfgang Tichy and Bernd Brügmann. "Quasiequilibrium binary black hole sequences for puncture data derived from helical Killing vector conditions". In: *Phys. Rev. D* 69 (2 2004), p. 024006. DOI: `10.1103/PhysRevD.69.024006`. URL: `https://link.aps.org/doi/10.1103/PhysRevD.69.024006`.

[112] LN Trefethen. "Approximation Theory and Approximation Practice (Philadelphia, PA, USA: Society for Industrial and Applied Mathematics)". In: (2012).

[113] Shuji Uchida and Tatsuo Yoshida. "Self-similar collapse with cooling and heating in an expanding universe". In: MNRAS 348.1 (Feb. 2004), pp. 89–99. DOI: `10.1111/j.1365-2966.2004.07328.x`. arXiv: `astro-ph/0311116 [astro-ph]`.

[114] Jean-Philippe Uzan. "Varying Constants, Gravitation and Cosmology". In: *Living Reviews in Relativity* 14.1 (2011), p. 2. ISSN: 1433-8351. DOI: `10.12942/lrr-2011-2`. URL: `https://doi.org/10.12942/lrr-2011-2`.

[115] A.I. Vainshtein. "To the problem of nonvanishing gravitation mass". In: *Phys. Lett. B* 39 (1972), pp. 393–394. DOI: `10.1016/0370-2693(72)90147-5`.

[116] Steven Weinberg. "The cosmological constant problem". In: *Rev. Mod. Phys.* 61 (1 1989), pp. 1–23. DOI: `10.1103/RevModPhys.61.1`. URL: `https://link.aps.org/doi/10.1103/RevModPhys.61.1`.

[117] T. Weinzierl et al. "Two particle-in-grid realisations on spacetrees". In: *Parallel Computing* 52 (2016), pp. 42–64. ISSN: 0167-8191. DOI: `https://doi.org/10.1016/j.parco.2015.12.007`. URL: `https://www.sciencedirect.com/science/article/pii/S0167819115001635`.

[118] Tobias Weinzierl. "The Peano Software-Parallel, Automaton-Based, Dynamically Adaptive Grid Traversals". In: *ACM Trans. Math. Softw.* 45.2 (Apr. 2019). ISSN: 0098-3500. DOI: `10.1145/3319797`. URL: `https://doi.org/10.1145/3319797`.

[119] Helvi Witek et al. "Black holes and binary mergers in scalar Gauss-Bonnet gravity: Scalar field dynamics". In: *Phys. Rev. D* 99 (6 2019), p. 064035. DOI: `10.1103/PhysRevD.99.064035`. URL: `https://link.aps.org/doi/10.1103/PhysRevD.99.064035`.

[120] R. Womersley. *Efficient Spherical Designs with Good Geometric Properties.* Website. Visited on Dec. 23, 2015. 2015. URL: `http://web.maths.unsw.edu.au/~rsw/Sphere/EffSphDes/index.html`.

[121] Hwei-Jang Yo, Thomas W. Baumgarte, and Stuart L. Shapiro. "Improved numerical stability of stationary black hole evolution calculations". In: *Phys. Rev. D* 66 (8 2002), p. 084026. DOI: `10.1103/PhysRevD.66.084026`. URL: `https://link.aps.org/doi/10.1103/PhysRevD.66.084026`.

APPENDIX A

Spherical Symmetry of the Simulated Solution in the Spherical Collapse Application

In this appendix, we show that the simulation results from our Spherical Collapse application are highly close to exact spherical symmetry during the stable evolution phase. In Figure A.1, we plot the rescaled profiles of the same three physical quantities discussed in Section 3.3, sampled along six different directions (shown in different colours) from the same simulation. The sampling directions are all on the x-y plane from a slice of the simulation box perpendicular to the z-axis. The coordinates shown in the legend are the starting and ending points of the sampling axis, while the black dashed line is the theoretical self-similar prediction plotted for comparison.

These profiles all agree with each other nearly perfectly except in the small region immediately inside the shock in the velocity profile (middle) panel, where the curves in different directions deviate from each other slightly and the profile in the diagonal direction (brown line, $(0, 0)$–$(1.5, 1.5)$) shows the most similar shape to the theoretical pattern (though the simulation result has a different amplitude due the reason explained in the main text). In particular, we note that the shock position is in good agreement along the different directions.

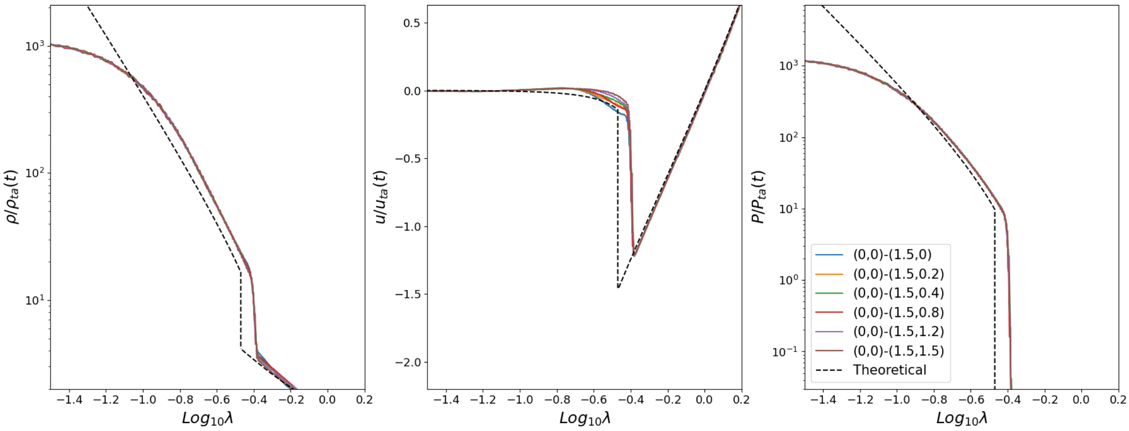

Figure A.1: The rescaled profiles of physical quantities in six different directions (as given in the legend) from the same simulation. It shows that the profiles of all considered quantities only have a very weak dependence on the direction along which we sample the solutions. See the text for more details.

APPENDIX B

Detailed Derivation of CCZ4 Evolving Formulation in Numerical Relativity

In this appendix, we provide a thorough derivation of the evolving equation we introduced in sections 4.1.3 and 4.5. The notation we used here is consistent with the ones in chapter 4.

B.1 The Standard Gravity

We show how we derive the evolving system (4.71) - (4.83) from the original field equation (4.50) in this section. The derivation of the ADM formulation (4.24) - (4.29) is not included here. However, it can be found in any standard Numerical Relativity textbook (e.g., Baumgarte and Shapiro, 2010). We will also use the following Hamilton and Momentum constraints of the Z4 system, which can be derived using a similar approach to the derivation of (4.17) and (4.18):

$$\begin{aligned} R + K^2 - K_{ab}K^{ab} &= 2n^p n^r [{}^{(4)}R_{pr} - \frac{1}{2} g^{(4)}_{pr} R] \\ &= 16\pi\rho + 2\kappa_1[2\Theta + \kappa_2\Theta] - 2n^a n^b \nabla_a Z_b - 2n^a n^b \nabla_b Z_a - 2\nabla^c Z_c, \end{aligned} \tag{B.1.1}$$

$$\begin{aligned} D_b K^b_c - D_c K &= -\gamma^p_c n^r [{}^{(4)}R_{pr} - \frac{1}{2} g^{(4)}_{pr} R] \\ &= 8\pi S_c + \kappa_1 \gamma^a_c Z_a + \gamma^a_c n^b \nabla_a Z_b + \gamma^a_c n^b \nabla_b Z_a. \end{aligned} \tag{B.1.2}$$

B.1.1 From Field Equation to Z4

Our first step is to get the Z4 system (4.52) - (4.55). The evolving equation of γ_{ij} (4.52) is the same as it is in the ADM formulation (4.26), thus no work is needed. We then check the evolving equation of K_{ij}. We start the derivation from its Lie derivative over $\mathbf{n}$:

$$\begin{aligned} \mathcal{L}_{\mathbf{n}} K_{ab} &= n^d n^c \gamma^q_a \gamma^{r(4)}_b R_{drcq} - \frac{1}{\alpha} D_a D_b \alpha - K^c_b K_{ac} \\ &= -\frac{1}{\alpha} D_a D_b \alpha - K^c_b K_{ac} + \gamma^{cd} \gamma^q_a \gamma^{r(4)}_b R_{drcq} - \gamma^q_a \gamma^{r(4)}_b R_{rq} \\ &= -\frac{1}{\alpha} D_a D_b \alpha - K^c_b K_{ac} + R_{ab} + K K_{ab} - K_{ac} K^c_b \\ &\quad - \gamma^q_a \gamma^r_b [8\pi (T_{rq} - \frac{1}{2} g_{rq} T) - \nabla_r Z_q - \nabla_q Z_r + \kappa_1 (n_r Z_q + n_q Z_r - (1+\kappa_2) g_{rq} n_c Z^c)] \\ &= -\frac{1}{\alpha} D_a D_b \alpha + R_{ab} + K K_{ab} - 2 K_{ac} K^c_b - 8\pi (S_{ab} - \frac{1}{2}(S - \rho)\gamma_{ab}) \\ &\quad + 2\gamma^c_a \gamma^d_b \nabla_{(c} Z_{d)} - \kappa_1 (\kappa_2 + 1) \gamma_{ab} \Theta. \end{aligned} \tag{B.1.3}$$

We have

$$\begin{aligned} 2\gamma^c_a \gamma^d_b \nabla_{(c} Z_{d)} &= 2\gamma^c_a \gamma^d_b \nabla_c Z_d \\ &= \gamma^e_b \gamma^c_a \gamma^d_e \nabla_c Z_d + \gamma^e_a \gamma^c_e \gamma^d_b \nabla_c Z_d \\ &= [\gamma^e_b \gamma^c_a \nabla_c (\gamma^d_e Z_d) - \gamma^e_b \gamma^c_a Z_d \nabla_c \gamma^d_e] + (a \leftrightarrow b) \\ &= \gamma^e_b \gamma^c_a \nabla_c (\gamma^d_e Z_d) - \gamma^e_b \gamma^c_a Z_d n^d \nabla_c n_e - \cancel{\gamma^e_b \gamma^c_a Z_d n_e \nabla n^d} + (a \leftrightarrow b) \\ &= \gamma^e_b \gamma^c_a \nabla_c (\gamma^d_e Z_d) - \Theta K_{ab} + (a \leftrightarrow b). \end{aligned} \tag{B.1.4}$$

Its corresponding spatial expression is $\nabla_i Z_j + \nabla_j Z_i - 2\Theta K_{ij}$, then simply use $\mathcal{L}_{\mathbf{t}} = \alpha \mathcal{L}_{\mathbf{n}} + \mathcal{L}_{\beta}$ we get the equation (4.53).

We use a similar approach for the evolving equation of Θ:

$$\begin{aligned}
\mathcal{L}_{\mathbf{n}}\Theta &= n^a\nabla_a\Theta \\
&= n^a\nabla_a(-n^bZ_b) \\
&= -n^an^b\nabla_aZ_b - n^aZ_b\nabla_an^b \\
&= n^an^b\nabla_bZ_a + \nabla^cZ_c + \frac{1}{2}[R - K_{ab}K^{ab} + K^2 - 16\pi\rho - 2\kappa_1(2+\kappa_2)\Theta] - n^aZ_b\nabla_an^b \\
&= n^an^b\nabla_bZ_a - n^aZ_b\nabla_an^b + [\gamma^{ab}\gamma^c_a\gamma^d_b\nabla_c(\gamma^e_dZ_e) - n^cn^e\nabla_cZ_e - n^eZ_e\nabla_an^a] \\
&\quad + \frac{1}{2}[R - K_{ab}K^{ab} + K^2 - 16\pi\rho - 2\kappa_1(2+\kappa_2)\Theta] \\
&= \frac{1}{2}[R + 2\gamma^{ab}\gamma^c_a\gamma^d_b\nabla_c(\gamma^e_dZ_e) + K^2 - K_{ab}K^{ab} - 2\kappa_1(2+\kappa_2)\Theta - 16\pi\rho] \\
&\quad - n^eZ_e\nabla_an^a - n^aZ_b\nabla_an^b \\
&= \frac{1}{2}[R + 2\gamma^{ab}\gamma^c_a\gamma^d_b\nabla_c(\gamma^e_dZ_e) + K^2 - K_{ab}K^{ab} - 2\kappa_1(2+\kappa_2)\Theta - 16\pi\rho] \\
&\quad - n^eZ_e\nabla_an^a - n^aZ_aK + \cancel{n^an_bZ^bn^c\nabla_cn_a} - \Theta K - \gamma^a_bZ^b(D_a\ln\alpha) \\
&= \frac{1}{2}[R + 2\gamma^{ab}\gamma^c_a\gamma^d_b\nabla_c(\gamma^e_dZ_e) + (K-2\Theta)K - K_{ab}K^{ab} - 2\gamma^a_bZ^b\nabla_a\alpha/\alpha - 2\kappa_1(2+\kappa_2)\Theta - 16\pi\rho] \\
&\quad \cancel{-n^eZ_e\nabla_an^a + n^aZ_a\nabla_en^e} + \cancel{n^aZ_an^cn^e\nabla_en_c},
\end{aligned} \tag{B.1.5}$$

Then we get the equation (4.54). The 4th equal sign uses the Hamiltonian constraint (B.1.1). The 5th equal sign uses the relation:

$$\begin{aligned}
\gamma^{ab}\gamma^c_a\gamma^d_b\nabla_c(\gamma^e_dZ_e) &= \gamma^{ab}[\gamma^c_a\gamma^e_b\nabla_cZ_e + \gamma^c_a\gamma^d_bZ_en^e\nabla_cn_d + \cancel{\gamma^c_a\gamma^d_bZ_en_d\nabla_cn^e}] \\
&= \gamma^{ab}[\nabla_aZ_b + n^en_b\nabla_aZ_e + n^cn_a\nabla_cZ_b + n^cn^en_an_b\nabla_cZ_e \\
&\quad + n^eZ_e\nabla_an_b + n^eZ_en^cn_a\nabla_cn_b + \cancel{n^eZ_en^dn_b\nabla_an_d} + \cancel{n^eZ_en^cn_an^dn_b\nabla_cn_d}] \\
&= \nabla_aZ^a + n^cn^e\nabla_cZ_e + n^eZ_e\nabla_an^a.
\end{aligned} \tag{B.1.6}$$

The 7th equal sign use relation:

$$\begin{aligned}
-\Theta K - \gamma^a_bZ^bD_a\ln\alpha &= -\Theta K - \gamma^a_bZ^bn^c\nabla_cn_a \\
&= n^aZ_aK - Z_an^c\nabla_cn^a - n^an_bZ^bn^c\nabla_cn_a,
\end{aligned} \tag{B.1.7}$$

where we use another relation $n^b\nabla_bn_a = D_a\ln\alpha$.

The evolving equation of Z_i is the spatial part of its counterpart for $Z_a = \gamma^b_aZ_b$, which is derived as:

$$\begin{aligned}\mathcal{L}_{\mathbf{n}}(\gamma_a^b Z_b) &= n^c\nabla_c(\gamma_a^b Z_b) + \gamma_c^b Z_b \nabla_a n^c \\ &= n^c Z_b \nabla_c \gamma_a^b + n^c \gamma_a^b \nabla_c Z_b + Z_b \nabla_a n^b \\ &= n^c Z_b \nabla_c \gamma_a^b + Z_b \nabla_a n^b + D_b K_a^b - D_a K - 8\pi S_a - \kappa_1 \gamma_a^c Z_c - \gamma_a^c n^b \nabla_c Z_b \\ &= D_b K_a^b - D_a K - 8\pi S_a - \kappa_1 \gamma_a^c Z_c \\ &\quad + [n^c Z_b n^b \nabla_c n_a + Z_b \nabla_a n^b - n^b \nabla_a Z_b - n^c n^b n_a \nabla_c Z_b + n^c Z_b n_a \nabla_c n^b].\end{aligned} \tag{B.1.8}$$

The 3rd equal sign uses the Momentum constraint (B.1.2). On the other hand, we have

$$\begin{aligned}\gamma_a^b \nabla_b \Theta - 2K_a^b \gamma_b^c Z_c + n^b Z_b n^c \nabla_c n_a &\\ &= -\gamma_a^b \nabla_b (n^c Z_c) + 2(\nabla_a n^b + n_a n^d \nabla_d n^b)\gamma_b^c Z_c + n^b Z_b n^c \nabla_c n_a \\ &= -n^c \nabla_a Z_c - Z_c \nabla_a n^c - n^b n_a \nabla_b (n^c Z_c) \\ &\qquad + 2Z_b \nabla_a n^b + 2 Z_b n_a n^d \nabla_d n^b + n^b Z_b n^c \nabla_c n_a \\ &= [n^c Z_b n^b \nabla_c n_a + Z_b \nabla_a n^b - n^b \nabla_a Z_b \\ &\qquad + n^c Z_b n_a \nabla_c n^b - n^b n^c n_a \nabla_b Z_c].\end{aligned} \tag{B.1.9}$$

This is equal to the second line of the last step of (B.1.8).

$$\begin{aligned}\mathcal{L}_{\mathbf{n}}(\gamma_a^b Z_b) &= D_b K_a^b - D_a K - 8\pi S_a - \kappa_1 \gamma_a^c Z_c \\ &\quad + \gamma_a^b \nabla_b \Theta - 2K_a^b \gamma_b^c Z_c + n^b Z_b n^c \nabla_c n_a.\end{aligned} \tag{B.1.10}$$

We also have

$$\gamma_a^b \nabla_b \Theta \to \partial_i \Theta, \quad n^b Z_b n^c \nabla_c n_a \to -\Theta D_a \ln\alpha \to -\Theta \nabla_i \alpha/\alpha, \tag{B.1.11}$$

when moving to the spatial part of the equation. It finally leads us to the equation (4.55).

B.1.2 From Z4 to FOCCZ4

Our next step is to derive the CCZ4 system (4.59) - (4.64) based on the conformal decomposition (4.56), (4.57) and the new evolving variable $\hat{\Gamma}^i$ (4.58). Following that, We introduce the auxiliary variables (4.68) in our system to get the FOCCZ4 equations utilised in our application.

We start by checking the evolving equation of conformal factor ϕ. Notice

$$\begin{aligned}\partial_t\gamma &= \gamma\gamma^{ij}\partial_t\gamma_{ij} \\ &= \gamma\gamma^{ij}(-2\alpha K_{ij} + D_i\beta_j + D_j\beta_i + \beta^k D_k\gamma_{ij}) \\ &= -2\alpha K\gamma + 2\gamma D_i\beta^i + \beta^k D_k\gamma,\end{aligned} \tag{B.1.12}$$

and $\gamma = \phi^{-6}$, we immediately get

$$\partial_t\phi = \frac{1}{3}\alpha K\phi - \frac{1}{3}\phi\partial_k\beta^k + \beta^k\partial_k\phi, \tag{B.1.13}$$

which is equation (4.61).

The evolving equation of K is given as

$$\begin{aligned}(\partial_t - \mathcal{L}_\beta)K &= (\partial_t - \mathcal{L}_\beta)[\gamma^{ij}K_{ij}] \\ &= K_{ij}(\partial_t - \mathcal{L}_\beta)(\gamma^{ij}) + \gamma^{ij}(\partial_t - \mathcal{L}_\beta)K_{ij}.\end{aligned} \tag{B.1.14}$$

By inserting the equation of γ^{ij}, which is the lifted case of (4.52), and K_{ij}, we can derive the equation (4.62).

Similarly, we have

$$\begin{aligned}(\partial_t - \mathcal{L}_\beta)\tilde{\gamma}_{ij} &= (\partial_t - \mathcal{L}_\beta)(\phi^2\gamma_{ij}) \\ &= \phi^2(\partial_t - \mathcal{L}_\beta)\gamma_{ij} + 2\gamma_{ij}\phi(\partial_t - \mathcal{L}_\beta)\phi\end{aligned} \tag{B.1.15}$$

for $\tilde{\gamma}_{ij}$, use equation (4.52) and (4.61) we get equation (4.59).

The evolution equation of Θ in the CCZ4 system (4.63) is nearly the same as it is in the Z4 system (4.54), the only term that gets changed is the expression of contraction of K_{ij}:

$$K^{ij}K_{ij} = (\phi^{-2}\tilde{A}_{ij} + \frac{1}{3}K\gamma_{ij})(\phi^2\tilde{A}^{ij} + \frac{1}{3}K\gamma^{ij}) = \tilde{A}^{ij}\tilde{A}_{ij} + \frac{1}{3}K^2. \tag{B.1.16}$$

For evolution of $\tilde{A}_{ij}$, we have

$$\begin{aligned}(\partial_t - \mathcal{L}_\beta)\tilde{A}_{ij} &= (\partial_t - \mathcal{L}_\beta)\phi^2(K_{ij} - \frac{1}{3}K\gamma_{ij}) \\ &= 2\phi(K_{ij} - \frac{1}{3}K\gamma_{ij})(\partial_t - \mathcal{L}_\beta)\phi + \phi^2(\partial_t - \mathcal{L}_\beta)K_{ij} \\ &\quad - \frac{1}{3}\phi^2\gamma_{ij}(\partial_t - \mathcal{L}_\beta)K - \frac{1}{3}\phi^2 K(\partial_t - \mathcal{L}_\beta)\gamma_{ij}.\end{aligned} \tag{B.1.17}$$

Use equations (4.61), (4.53), (4.62) and (4.52) and performing a series of intricate calculations, we arrive at (4.60).

For evolution of $\hat{\Gamma}^i$, we have

$$\begin{aligned}\partial_t\hat{\Gamma}^i &= \partial_t(-\partial_j\tilde{\gamma}^{ij} + 2\tilde{\gamma}^{ij}Z_j) \\ &= -\partial_j\partial_t\tilde{\gamma}^{ij} + 2Z_j\partial_t\tilde{\gamma}^{ij} + 2\tilde{\gamma}^{ij}\partial_t Z_j.\end{aligned} \tag{B.1.18}$$

Again the usage of equations (4.59) and (4.55) gives us the equation (4.64). Most of the terms are just merged and recast in the new notation, but there are two terms that get rewritten and simplified:

$$-2\alpha\partial_j\tilde{A}^{ij} + 2\alpha\tilde{\gamma}^{ij}\nabla_l\tilde{A}^l_j \equiv 2\alpha(\tilde{\Gamma}^i_{jk}\tilde{A}^{jk} - 3\tilde{A}^{ij}\frac{\partial_j\phi}{\phi}). \tag{B.1.19}$$

We prove this simplification as follows: first, one may notice that

$$\begin{aligned}-2\alpha\partial_j\tilde{A}^{ij} + 2\alpha\tilde{\gamma}^{ij}\nabla_l\tilde{A}^l_j &= -2\alpha\partial_j\tilde{A}^{ij} + 2\alpha\tilde{\gamma}^{ij}(\partial_l\tilde{A}^l_j + \Gamma^l_{ml}\tilde{A}^m_j - \Gamma^l_{jm}\tilde{A}^m_l) \\ &= -2\alpha\tilde{A}^{jk}\tilde{\gamma}_{mk}\partial_j\tilde{\gamma}^{im} + 2\alpha\tilde{\gamma}^{ij}(\Gamma^l_{ml}\tilde{A}^m_j - \Gamma^l_{jm}\tilde{A}^m_l).\end{aligned} \tag{B.1.20}$$

On the other hand, we have

$$\begin{aligned}2\alpha\tilde{\Gamma}^i_{jk}\tilde{A}^{jk} &= \alpha\tilde{A}^{jk}\tilde{\gamma}^{im}(\partial_j\tilde{\gamma}_{mk} + \partial_k\tilde{\gamma}_{mj} - \partial_m\tilde{\gamma}_{jk}) \\ &= -2\alpha\tilde{A}^{jk}\tilde{\gamma}_{mk}\partial_j\tilde{\gamma}^{im} - \alpha\tilde{A}^{jk}\tilde{\gamma}^{im}\partial_m\tilde{\gamma}_{jk}.\end{aligned} \tag{B.1.21}$$

Thus we get

$$
\begin{aligned}
-2\alpha\partial_j\tilde{A}^{ij} + 2\alpha\tilde{\gamma}^{ij}\nabla_l\tilde{A}^l_j &= 2\alpha\tilde{\Gamma}^i_{jk}\tilde{A}^{jk} + \alpha\tilde{A}^{jk}\tilde{\gamma}^{im}\partial_m\tilde{\gamma}_{jk} + 2\alpha\tilde{\gamma}^{ij}(\Gamma^l_{ml}\tilde{A}^m_j - \Gamma^l_{jm}\tilde{A}^m_l) \\
&= 2\alpha\tilde{\Gamma}^i_{jk}\tilde{A}^{jk} + \alpha\tilde{A}^{jk}\tilde{\gamma}^{im}\partial_m\tilde{\gamma}_{jk} + 2\alpha\tilde{A}^{mi}(\tilde{\Gamma}^l_{ml} - 3\frac{\partial_m\phi}{\phi}) \\
&\quad - 2\alpha\tilde{A}^m_l\tilde{\gamma}^{ij}(\tilde{\Gamma}^l_{jm} - \delta^l_j\frac{\partial_m\phi}{\phi} - \delta^l_m\frac{\partial_j\phi}{\phi} + \tilde{\gamma}_{jm}\tilde{\gamma}^{ln}\frac{\partial_n\phi}{\phi}) \\
&= 2\alpha\tilde{\Gamma}^i_{jk}\tilde{A}^{jk} + \alpha\tilde{A}^{jk}\tilde{\gamma}^{im}\partial_m\tilde{\gamma}_{jk} + 2\alpha\tilde{A}^{mi}(\tilde{\Gamma}^l_{ml} - 3\frac{\partial_m\phi}{\phi}) \\
&\quad - 2\alpha\tilde{A}^m_l(\tilde{\gamma}^{ij}\tilde{\Gamma}^l_{jm}\cancel{-\tilde{\gamma}^{il}\frac{\partial_m\phi}{\phi} + \delta^i_m\tilde{\gamma}^{ln}\frac{\partial_n\phi}{\phi}}) \\
&= 2\alpha(\tilde{\Gamma}^i_{jk}\tilde{A}^{jk} - 3\tilde{A}^{ij}\frac{\partial_j\phi}{\phi}) \\
&\quad + \alpha\tilde{A}^{jk}\tilde{\gamma}^{im}\partial_m\tilde{\gamma}_{jk} + 2\alpha\tilde{A}^{mi}\tilde{\Gamma}^l_{ml} - 2\alpha\tilde{A}^m_l\tilde{\gamma}^{ij}\tilde{\Gamma}^l_{jm}.
\end{aligned}
\tag{B.1.22}
$$

the second equality uses the relation (it slightly deviates from the one we introduced in section 4.1.2 as the definition of conformal factor is different):

$$
\Gamma^i_{jk} = \tilde{\Gamma}^i_{jk} - \delta^i_j\frac{\partial_k\phi}{\phi} - \delta^i_k\frac{\partial_j\phi}{\phi} + \tilde{\gamma}_{jk}\tilde{\gamma}^{il}\frac{\partial_l\phi}{\phi}. \tag{B.1.23}
$$

A further examination shows that

$$
\begin{aligned}
&\alpha\tilde{A}^{jk}\tilde{\gamma}^{im}\partial_m\tilde{\gamma}_{jk} + 2\alpha\tilde{A}^{mi}\tilde{\Gamma}^l_{ml} - 2\alpha\tilde{A}^m_l\tilde{\gamma}^{ij}\tilde{\Gamma}^l_{jm} \\
&\quad =\alpha\tilde{A}^{jk}\tilde{\gamma}^{im}\partial_m\tilde{\gamma}_{jk} + \alpha[\tilde{A}^{mi}\tilde{\gamma}^{lk}(\cancel{\partial_m\tilde{\gamma}_{kl}} + \cancel{\partial_l\tilde{\gamma}_{km} - \partial_k\tilde{\gamma}_{ml}}) - \tilde{A}^{mk}\tilde{\gamma}^{ij}(\partial_j\tilde{\gamma}_{km} + \cancel{\partial_m\tilde{\gamma}_{kj} - \partial_k\tilde{\gamma}_{jm}})] \\
&\quad =\alpha\tilde{A}^{jk}\tilde{\gamma}^{im}\partial_m\tilde{\gamma}_{jk} - \alpha\tilde{A}^{mk}\tilde{\gamma}^{ij}\partial_j\tilde{\gamma}_{km} \\
&\quad =0.
\end{aligned}
\tag{B.1.24}
$$

Therefore the equation (B.1.22) is identical to the simplification we show in equation (B.1.19).

Introducing the auxiliary variables into the evolving equations of primary variables (4.71) - (4.79) is straightforward, we simply replace the spatial derivatives of α, β^i, $\tilde{\gamma}_{ij}$ and ϕ with the corresponding variables in (4.68). The evolving equations of the auxiliary variables themselves are derived using the commutativity of the derivatives:

$$\partial_t A_k = \partial_t \partial_i \alpha = \partial_i(\partial_t \alpha). \tag{B.1.25}$$

We then put the equation (4.72) into it and get (4.80). The equations (4.81) - (4.83) can be calculated in a similar approach.

The relations (4.84) - (4.94) we used in our FOCCZ4 system can be derived from their definition straightforwardly, the only relation in a little tricky is (4.84):

$$\begin{aligned}
\partial_k \tilde{\gamma}^{ij} &= \delta^i_m \partial_k \tilde{\gamma}^{mj} \\
&= \tilde{\gamma}^{in} \tilde{\gamma}_{nm} \partial_k \tilde{\gamma}^{mj} \\
&= -\tilde{\gamma}^{in} \tilde{\gamma}^{mj} \partial_k \tilde{\gamma}_{nm} \\
&= -2\tilde{\gamma}^{in} \tilde{\gamma}^{mj} D_{knm} \\
&:= -2D^{ij}_k,
\end{aligned} \tag{B.1.26}$$

where we use

$$\tilde{\gamma}_{nm} \partial_k \tilde{\gamma}^{mj} + \tilde{\gamma}^{mj} \partial_k \tilde{\gamma}_{nm} = \partial_k(\tilde{\gamma}_{nm} \tilde{\gamma}^{mj}) = \partial_k \delta^j_n = 0. \tag{B.1.27}$$

B.2 The Scalar Field Theory

We then give the derivation of the evolving system (4.174) - (4.184) under the scalar field gravity, starting from the equations (4.156) - (4.157). Throughout the calculation, the two constraints (4.159) - (4.160) presented in the content will also be utilised.

B.2.1 From Field Equation"s" to Z4

Again, we first derive the Z4 formulation (4.162) - (4.167) for the scalar field gravity.

We first notice that the equation for γ_{ij} (4.162) is identical to the one in the standard gravity and the equation for φ (4.163) is from the definition (4.161), so no derivation is needed for these two equations.

The evolving equation of K_{ab} is again from its Lie derivative along $\mathbf{n}$:

$$\begin{aligned}
\mathcal{L}_{\mathbf{n}}K_{ab} &= -\frac{1}{\alpha}D_aD_b\alpha - 2K^c_aK_{bc} + R_{ab} + KK_{ab} - \gamma^c_a\gamma^d_b{}^{(4)}R_{cd} \\
&= -\frac{1}{\alpha}D_aD_b\alpha - 2K^c_aK_{bc} + R_{ab} + KK_{ab} \\
&\quad - F^{-1}(\varphi)\gamma^c_a\gamma^d_b\Bigg[8\pi(T_{cd} - \frac{1}{2}g_{cd}T) + Z(\varphi)\nabla_c\varphi\nabla_d\varphi + \nabla_c\nabla_dF(\varphi) \\
&\qquad + \frac{1}{2}g_{cd}\nabla^e\nabla_eF(\varphi) + g_{cd}U(\varphi) - \nabla_cZ_d - \nabla_dZ_c \\
&\qquad + \kappa_1\left[n_cZ_d + n_dZ_c - (1+\kappa_2)g_{cd}n_eZ^e\right]\Bigg] \\
&= -\frac{1}{\alpha}D_aD_b\alpha - 2K^c_aK_{bc} + R_{ab} + KK_{ab} - F^{-1}(\varphi)8\pi\left(S_{ab} - \frac{1}{2}(S-\rho)\gamma_{ab}\right) \\
&\quad + F^{-1}(\varphi)(\gamma^e_b\gamma^c_a\nabla_c(\gamma^d_eZ_d) + \gamma^e_a\gamma^c_b\nabla_c(\gamma^d_eZ_d)) - F^{-1}(\varphi)2\Theta K_{ab} \\
&\quad - F^{-1}(\varphi)\kappa_1(1+\kappa_2)\gamma_{ab}\Theta - \gamma_{ab}F^{-1}(\varphi)U(\varphi) \\
&\quad - F^{-1}(\varphi)\gamma^c_a\gamma^d_b\Bigg[Z(\varphi)\nabla_c\varphi\nabla_d\varphi + \nabla_c\nabla_dF(\varphi) + \frac{1}{2}g_{cd}\nabla^e\nabla_eF(\varphi)\Bigg]
\end{aligned} \tag{B.2.28}$$

The first three lines could be translated into spatial expressions easily. We then check the final line:

$$\begin{aligned}
&-F^{-1}(\varphi)\gamma^c_a\gamma^d_b\Bigg[Z(\varphi)\nabla_c\varphi\nabla_d\varphi + \nabla_c\nabla_dF(\varphi) + \frac{1}{2}g_{cd}\nabla^c\nabla_cF(\varphi)\Bigg] \\
&\quad = -\frac{F''(\varphi) + Z(\varphi)}{F(\varphi)}(\gamma^c_a\nabla_c\varphi)(\gamma^d_b\nabla_d\varphi) \\
&\qquad - F^{-1}(\varphi)\gamma^c_a\gamma^d_b\Bigg[F'(\varphi)\nabla_c\nabla_d\varphi + \frac{1}{2}g_{cd}(F''(\varphi)\nabla^e\varphi\nabla_e\varphi + F'(\varphi)\nabla^e\nabla_e\varphi)\Bigg] \\
&\quad = -\frac{F''(\varphi) + Z(\varphi)}{F(\varphi)}(\gamma^c_a\nabla_c\varphi)(\gamma^d_b\nabla_v\varphi) - F^{-1}(\varphi)F'(\varphi)[\gamma^c_a\gamma^d_b\nabla_c(\gamma^e_d\nabla_e\varphi) - K_\varphi K_{ab}] \\
&\qquad - F^{-1}(\varphi)\frac{1}{2}\gamma_{ab}\Bigg[F''(\varphi)(\gamma^{ef}(\gamma^c_e\nabla_c\varphi)(\gamma^d_f\nabla_d\varphi) - K^2_\varphi) \\
&\qquad\qquad + F'(\varphi)(\gamma^{ef}\gamma^c_e\gamma^d_f\nabla_c(\gamma^g_d\nabla_g\varphi) + \alpha^{-1}(D^e\alpha)(D_e\varphi) - KK_\varphi + \mathcal{L}_{\mathbf{n}}K_\varphi)\Bigg]
\end{aligned} \tag{B.2.29}$$

One can see the evolution of K_{ab} is coupled with K_φ now. The first equality uses relation:

$$\nabla_a\nabla_bF(\varphi) = F''\nabla_a\varphi\nabla_b\varphi + F'\nabla_a\nabla_b\varphi. \tag{B.2.30}$$

The second equality uses relations:

$$\begin{aligned}
\gamma^b_e\gamma^c_d\nabla_c(\gamma^a_b\nabla_a\varphi) &= \gamma^a_e\gamma^c_d\nabla_c\nabla_a\varphi + \gamma^b_e\gamma^c_d\nabla_a\varphi(\nabla_c\gamma^a_b)\\
&= \gamma^a_e\gamma^c_d\nabla_c\nabla_a\varphi + n^a\nabla_a\varphi\gamma^b_e\gamma^c_d\nabla_c n_b\\
&= \gamma^a_e\gamma^c_d\nabla_c\nabla_a\varphi + K_\varphi K_{ed} \qquad \text{(B.2.31)}\\
&\Downarrow\\
\gamma^c_a\gamma^d_b\nabla_c\nabla_d\varphi &= \gamma^c_a\gamma^d_b\nabla_c(\gamma^e_d\nabla_e\varphi) - K_\varphi K_{ab},
\end{aligned}$$

$$\begin{aligned}
\gamma^{ab}(\gamma^c_a\nabla_c\varphi)(\gamma^d_b\nabla_d\varphi) &= g^{ab}\nabla_a\varphi\nabla_b\varphi + g^{ab}(n_a n^c\nabla_c\varphi)(n_b n^d\nabla_d\varphi)\\
&\quad + g^{ab}(n_a n^c\nabla_c\varphi)(\nabla_b\varphi) + g^{ab}\nabla_a\varphi(n_b n^d\nabla_d\varphi)\\
&= g^{ab}\nabla_a\varphi\nabla_b\varphi + K^2_\varphi \qquad \text{(B.2.32)}\\
&\Downarrow\\
\nabla^c\varphi\nabla_c\varphi &= \gamma^{ab}(\gamma^c_a\nabla_c\varphi)(\gamma^d_b\nabla_d\varphi) - K^2_\varphi,
\end{aligned}$$

$$\begin{aligned}
\gamma^{ab}\gamma^c_a\gamma^d_b\nabla_c(\gamma^e_d\nabla_e\varphi) &= \gamma^{cd}\nabla_c(\gamma^e_d\nabla_e\varphi)\\
&= g^{cd}\nabla_c(\gamma^e_d\nabla_e\varphi) + n^c n^d\nabla_c(\gamma^e_d\nabla_e\varphi)\\
&= \gamma^{ce}\nabla_c\nabla_e\varphi + g^{cd}(\nabla_e\varphi)(\nabla_c\gamma^e_d) + n^c n^d\nabla_c(\gamma^e_d\nabla_e\varphi)\\
&= g^{ce}\nabla_c\nabla_e\varphi + n^c n^e\nabla_c\nabla_e\varphi + \gamma^{cd}(\nabla_e\varphi)\nabla_c(n^e n_d)\\
&= g^{ce}\nabla_c\nabla_e\varphi + n^c\nabla_c(n^e\nabla_e\varphi) - n^c(\nabla_c n^e)(\nabla_e\varphi) + n^e\nabla_e\varphi\gamma^{cd}\nabla_c n_d\\
&= g^{ce}\nabla_c\nabla_e\varphi - \mathcal{L}_{\mathbf{n}}K_\varphi - \alpha^{-1}(D^e\alpha)(D_e\varphi) + KK_\varphi\\
&\Downarrow\\
\nabla^c\nabla_c\varphi &= \gamma^{ef}\gamma^c_e\gamma^d_f\nabla_c(\gamma^g_d\nabla_g\varphi) + \alpha^{-1}(D^e\alpha)(D_e\varphi) - KK_\varphi + \mathcal{L}_{\mathbf{n}}K_\varphi.
\end{aligned}$$

$$\text{(B.2.33)}$$

The term $\mathcal{L}_{\mathbf{n}}K_\varphi$ is highlighted in red to track its appearance. The spatial counterpart of (B.2.28) gives equation (4.164).

We then check the evolving equation of K_φ. Notice that the $\mathcal{L}_{\mathbf{n}}K_\varphi$ appears on the right-hand side of relation (B.2.33), whose left-hand side also appears in the field equation (4.157). Thus we use the field equation as our starting point:

$$\begin{aligned}
2Z(\varphi)\nabla^a\nabla_a\varphi &= -F'(\varphi)[\gamma^{pr}\gamma^{qs\,(4)}R_{pqrs} - 2n^p n^{r\,(4)}R_{pr}] - Z'(\varphi)\nabla^a\varphi\nabla_a\varphi + 2U'(\varphi) \\
&= -F'(\varphi)[R + K^2 - K_{ab}K^{ab}] - Z'(\varphi)\nabla^a\varphi\nabla_a\varphi + 2U'(\varphi) \\
&\quad + 2\frac{F'(\varphi)}{F(\varphi)}n^a n^b\Big[-\nabla_a Z_b - \nabla_b Z_a + \kappa_1[n_a Z_b + n_b Z_a - (1+\kappa_2)g_{ab}n_a Z^a] \\
&\qquad\qquad + 8\pi(T_{ab} - \frac{1}{2}g_{ab}T) + Z(\varphi)\nabla_a\varphi\nabla_b\varphi + \nabla_a\nabla_b F(\varphi) + \frac{1}{2}g_{ab}\nabla^a\nabla_a F(\varphi) + g_{ab}U(\varphi)\Big] \\
&= -F'(\varphi)[R + K^2 - K_{ab}K^{ab}] - Z'(\varphi)\nabla^a\varphi\nabla_a\varphi + 2U'(\varphi) \\
&\quad + 2\frac{F'(\varphi)}{F(\varphi)}\Big[2\mathcal{L}_{\mathbf{n}}\Theta + 2\gamma^{ab}Z_b D_a \ln\alpha + \kappa_1(1-\kappa_2)\Theta + 4\pi(a + S) \\
&\qquad\qquad + Z(\varphi)K_\varphi^2 - U(\varphi) + F''(\varphi)K_\varphi^2 - F'(\varphi)\mathcal{L}_{\mathbf{n}}K_\varphi - F'(\varphi)\gamma^{ab}\nabla_b\varphi D_a\ln\alpha\Big] \\
&\quad - \frac{F'(\varphi)}{F(\varphi)}[F''(\varphi)\nabla^a\varphi\nabla_a\varphi + F'(\varphi)\nabla^a\nabla_a\varphi].
\end{aligned} \tag{B.2.34}$$

For the first equality we use

$$^{(4)}R = \gamma^{pr}\gamma^{qs\,(4)}R_{pqrs} - 2n^p n^{r\,(4)}R_{pr}. \tag{B.2.35}$$

For the second equality we use

$$\gamma^{pr}\gamma^{qs\,(4)}R_{pqrs} = R + K^2 - K_{ab}K^{ab}, \tag{B.2.36}$$

and the trace reversed field equation (4.158). For the third equality we use

$$\begin{aligned}
n^a n^b(\nabla_a Z_b + \nabla_b Z_a) &= 2n^a n^b\nabla_a Z_b \\
&= 2[n^a\nabla_a(n^b Z_b) - Z_b n^a\nabla_a n^b] \\
&= -2(\mathcal{L}_{\mathbf{n}}\Theta + \gamma^{ab}Z_b D_a\ln\alpha),
\end{aligned} \tag{B.2.37}$$

and

$$\begin{aligned}
n^a n^b\nabla_a\nabla_b F(\varphi) &= n^a n^b[F''(\varphi)\nabla_a\varphi\nabla_b\varphi + F'(\varphi)\nabla_a\nabla_b\varphi] \\
&= F''(\varphi)K_\varphi^2 + F'(\varphi)[n^a\nabla_a(n^b\nabla_b\varphi) - (n^a\nabla_a n^b)\nabla_b\varphi] \\
&= F''(\varphi)K_\varphi^2 - F'(\varphi)\mathcal{L}_{\mathbf{n}}K_\varphi - F'(\varphi)\gamma^{ab}\nabla_b\varphi D_a\ln\alpha.
\end{aligned} \tag{B.2.38}$$

Therefore, we write the field equation (4.157) as

$$\begin{aligned}\left[2Z(\varphi)+\frac{F'(\varphi)^2}{F(\varphi)}\right]\nabla^a\nabla_a\varphi+2\frac{F'(\varphi)^2}{F(\varphi)}\mathcal{L}_{\mathbf{n}}K_\varphi\\ =-F'(\varphi)[R+K^2-K_{ab}K^{ab}]+2U'(\varphi)\\ +2\frac{F'(\varphi)}{F(\varphi)}\Big[2\gamma^{ab}Z_bD_a\ln\alpha+\kappa_1(1-\kappa_2)\Theta+4\pi(\rho+S)+Z(\varphi)K_\varphi^2-U(\varphi)\\ +F''(\varphi)K_\varphi^2-F'(\varphi)\gamma^{ab}\nabla_b\varphi D_a\ln\alpha\Big]\\ -\left[Z'(\varphi)+\frac{F'(\varphi)F''(\varphi)}{F(\varphi)}\right]\nabla^a\varphi\nabla_a\varphi+4\frac{F'(\varphi)}{F(\varphi)}\mathcal{L}_{\mathbf{n}}\Theta,\end{aligned}\tag{B.2.39}$$

then use relations (B.2.32) and (B.2.33) to get:

$$\begin{aligned}\left[2Z(\varphi)+\frac{F'(\varphi)^2}{F(\varphi)}\right][D^cD_c\varphi+(D^c\ln\alpha)(D_c\varphi)-KK_\varphi+\mathcal{L}_{\mathbf{n}}K_\varphi]+2\frac{F'(\varphi)^2}{F(\varphi)}\mathcal{L}_{\mathbf{n}}K_\varphi\\ =-F'(\varphi)[R+K^2-K_{ab}K^{ab}]+2U'(\varphi)\\ +2\frac{F'(\varphi)}{F(\varphi)}\Big[2\hat{Z}^aD_a\ln\alpha+\kappa_1(1-\kappa_2)\Theta+4\pi(\rho+S)+Z(\varphi)K_\varphi^2-U(\varphi)\\ +F''(\varphi)K_\varphi^2-F'(\varphi)D^a\varphi D_a\ln\alpha\Big]\\ -\left[Z'(\varphi)+\frac{F'(\varphi)F''(\varphi)}{F(\varphi)}\right](D^c\varphi D_c\varphi-K_\varphi^2)+4\frac{F'(\varphi)}{F(\varphi)}\mathcal{L}_{\mathbf{n}}\Theta.\end{aligned}\tag{B.2.40}$$

We have rewritten some quantities using the notation of spatial covariant derivatives D_a and then define $\hat{Z}_a=\gamma_a^bZ_b$ as the spatial projection of Z_a vector. The relation above finally gives us:

$$\begin{aligned}
&\left[2Z(\varphi)+3\frac{F'(\varphi)^2}{F(\varphi)}\right]\mathcal{L}_{\mathbf{n}}K_\varphi\\
&\quad=-F'(\varphi)[R+K^2-K_{ab}K^{ab}]-\left[Z'(\varphi)+\frac{F'(\varphi)F''(\varphi)}{F(\varphi)}\right](D^c\varphi D_c\varphi-K_\varphi^2)\\
&\qquad-\left[2Z(\varphi)+\frac{F'(\varphi)^2}{F(\varphi)}\right][D^cD_c\varphi+(D^c\ln\alpha)(D_c\varphi)-KK_\varphi]+2U'(\varphi)\\
&\qquad+2\frac{F'(\varphi)}{F(\varphi)}\Bigg[2\hat{Z}^aD_a\ln\alpha+\kappa_1(1-\kappa_2)\Theta+4\pi(\rho+S)+Z(\varphi)K_\varphi^2-U(\varphi)\\
&\qquad\qquad+F''(\varphi)K_\varphi^2-F'(\varphi)D^a\varphi D_a\ln\alpha\Bigg]+4\frac{F'(\varphi)}{F(\varphi)}\mathcal{L}_{\mathbf{n}}\Theta\\
&\quad=-F'(\varphi)[R+K^2-K_{ab}K^{ab}]+\left[Z'(\varphi)+3\frac{F'(\varphi)F''(\varphi)}{F(\varphi)}+2\frac{F'(\varphi)Z(\varphi)}{F(\varphi)}\right]K_\varphi^2\\
&\qquad-\left[2Z(\varphi)+\frac{F'(\varphi)^2}{F(\varphi)}\right]D_aD^a\varphi-\left[Z'(\varphi)+\frac{F'(\varphi)F''(\varphi)}{F(\varphi)}\right]D_a\varphi D^a\varphi\\
&\qquad-\left[2Z(\varphi)+3\frac{F'(\varphi)^2}{F(\varphi)}\right]D_a\ln\alpha D^a\varphi+\left[2Z(\varphi)+\frac{F'(\varphi)^2}{F(\varphi)}\right]KK_\varphi+2U'(\varphi)-2\frac{F'(\varphi)}{F(\varphi)}U(\varphi)\\
&\qquad+2\frac{F'(\varphi)}{F(\varphi)}\left[2\hat{Z}^aD_a\ln\alpha+\kappa_1(1-\kappa_2)\Theta+4\pi(\rho+S)\right]+4\frac{F'(\varphi)}{F(\varphi)}\mathcal{L}_{\mathbf{n}}\Theta.
\end{aligned}\tag{B.2.41}$$

It contains the Lie derivative of Θ which will be given below. Once we put that back into this equation, it yields the final expression (4.165).

We follow a similar approach as we did in standard gravity for the evolving equation of Θ:

$$\begin{aligned}
\mathcal{L}_{\mathbf{n}}\Theta&=n^a\nabla_a\Theta\\
&=n^a\nabla_a(-n^bZ_b)\\
&=-n^an^b\nabla_aZ_b-n^aZ_b\nabla_an^b\\
&=n^an^b\nabla_bZ_a+\nabla^cZ_c+\frac{1}{2}F(\varphi)[R-K_{ab}K^{ab}+K^2]-8\pi\rho-\kappa_1(2+\kappa_2)\Theta-n^aZ_b\nabla_an^b\\
&\quad-\frac{1}{2}Z(\varphi)[2K_\varphi^2+\nabla_a\varphi\nabla^a\varphi]-n^an^b\nabla_a\nabla_bF(\varphi)-\nabla_a\nabla^aF(\varphi)-U(\varphi)\\
&=\frac{1}{2}F(\varphi)[R+K^2-K_{ab}K^{ab}]+D_a\hat{Z}^a-\Theta K-\hat{Z}^aD_a\ln\alpha-\kappa_1(2+\kappa_2)\Theta-8\pi\rho\\
&\quad-Z(\varphi)K_\varphi^2-U(\varphi)-\frac{1}{2}Z(\varphi)\nabla_a\varphi\nabla^a\varphi-F''(\varphi)\nabla_a\varphi\nabla^a\varphi-F'(\varphi)\nabla^a\nabla_a\varphi-n^an^b\nabla_a\nabla_bF(\varphi)\\
&=\frac{1}{2}F(\varphi)[R+K^2-K_{ab}K^{ab}]+D_a\hat{Z}^a-\Theta K-\hat{Z}^aD_a\ln\alpha-\kappa_1(2+\kappa_2)\Theta-8\pi\rho\\
&\quad-\frac{1}{2}Z(\varphi)K_\varphi^2-\left[\frac{1}{2}Z(\varphi)+F''(\varphi)\right]D^a\varphi D_a\varphi-F'(\varphi)D_aD^a\varphi+F'(\varphi)KK_\varphi-U(\varphi).
\end{aligned}\tag{B.2.42}$$

The 4th equality uses Hamiltonian constraint (4.159). We then find that the part of the evolution of Z_a is nearly the same as one in (B.1.5). So we simply use its result in 5th equality. In the 6th equality we use relation (B.2.33), (B.2.32) and (B.2.38). Replacing all spacetime indices with spatial indices, we get equation (4.166).

Similarly, we calculate the evolving equation of $\hat{Z}_a$ as

$$
\begin{aligned}
\mathcal{L}_{\mathbf{n}}\hat{Z}_a &= n^c\nabla_c(\gamma^b_a Z_b) + \gamma^b_c Z_b \nabla_a n^c \\
&= n^c Z_b \nabla_c \gamma^b_a + n^c \gamma^b_a \nabla_c Z_b + Z_b \nabla_a n^b \\
&= n^c Z_b \nabla_c \gamma^b_a + Z_b \nabla_a n^b + F(\varphi)(D_b K^b_a - D_a K) - 8\pi S_a - \kappa_1 \gamma^c_a Z_c - \gamma^c_a n^b \nabla_c Z_b \\
&\quad - Z(\varphi) K_\varphi \gamma^b_a \nabla_b \varphi + n^b \gamma^c_a \nabla_b \nabla_c F(\varphi) \\
&= F(\varphi)(D_b K^b_a - D_a K) - 8\pi S_a - \kappa_1 \hat{Z}_a + D_a \Theta - 2K^b_a \hat{Z}_b - \Theta D_a \ln\alpha \\
&\quad - [Z(\varphi) + F''(\varphi)] K_\varphi D_a \varphi - F'(\varphi) D_a K_\varphi + F'(\varphi) K^b_a D_b \varphi
\end{aligned}
\tag{B.2.43}
$$

Again we reuse the relation (B.1.9) in the 4th equality. We also use

$$
\begin{aligned}
n^b \gamma^c_a \nabla_b \nabla_c F(\varphi) &= n^b \gamma^c_a [F''(\varphi) \nabla_b \varphi \nabla_c \varphi + F'(\varphi) \nabla_b \nabla_c \varphi] \\
&= -F''(\varphi) K_\varphi D_a \varphi + F'(\varphi)[\gamma^c_a \nabla_c (n^b \nabla_b \varphi) - \nabla_b \varphi (\gamma^c_a \nabla_c n^b)] \\
&= -F''(\varphi) K_\varphi D_a \varphi - F'(\varphi) D_a K_\varphi - F'(\varphi) \nabla_b \varphi (\gamma^c_a g^{bd} \nabla_c n_d) \\
&= -F''(\varphi) K_\varphi D_a \varphi - F'(\varphi) D_a K_\varphi - F'(\varphi) \nabla_b \varphi (\gamma^c_a \gamma^{bd} \nabla_c n_d) \\
&= -F''(\varphi) K_\varphi D_a \varphi - F'(\varphi) D_a K_\varphi + F'(\varphi) D_b \varphi K^b_a .
\end{aligned}
\tag{B.2.44}
$$

By replacing $\hat{Z}_a \to Z_i$ we get the expression (4.167).

B.2.2 From Z4 to FOCCZ4

We now apply the conformal decomposition to our new evolving system under scalar field gravity. For now, The scalar field is assumed not subject to decomposition, but it may change in future investigations.

Compared to the results we get for the standard gravity, Two evolving equations remain the same, namely the ones of $\tilde{\gamma}_{ij}$ and ϕ. The equation of φ is still from the definition of K_φ:

$$\partial_t\varphi = -\alpha K_\varphi + \beta^k\partial_k\varphi. \tag{B.2.45}$$

For evolving equations of Θ, we again only need to change the expression of contraction of K_{ij} using relation (B.1.16). This evolving equation now read as

$$\begin{aligned}\partial_t\Theta = {}& \frac{\alpha}{2}F(\varphi)\left[R+\frac{2}{3}K^2-\tilde{A}_{ij}\tilde{A}^{ij}\right]+\alpha D_iZ^i-\alpha\Theta K-Z^iD_i\alpha-\alpha\kappa_1(2+\kappa_2)\Theta\\ &-8\pi\alpha\rho-\frac{1}{2}Z(\varphi)\alpha K_\varphi^2-\left[\frac{1}{2}Z(\varphi)+F''(\varphi)\right]\alpha D^i\varphi D_i\varphi-\alpha F'(\varphi)D_iD^i\varphi\\ &+\alpha F'(\varphi)KK_\varphi-\alpha U(\varphi)+\beta^k\partial_k\Theta.\end{aligned} \tag{B.2.46}$$

As the K_φ also remains untouched from the decomposition, we only need to replace K_{ij} with $\tilde{A}_{ij}$ in that content as well. It now gives

$$\begin{aligned}\partial_tK_\varphi = {}& \beta^i\partial_iK_\varphi+\alpha\left[2Z(\varphi)+3\frac{F'(\varphi)^2}{F(\varphi)}\right]^{-1}\\ &\Bigg\{F'(\varphi)[R+\frac{2}{3}K^2-\tilde{A}_{ij}\tilde{A}^{ij}]+\left[Z'(\varphi)+3\frac{F'(\varphi)F''(\varphi)}{F(\varphi)}\right]K_\varphi^2-\left[2Z(\varphi)+5\frac{F'(\varphi)^2}{F(\varphi)}\right]D_iD^i\varphi\\ &-\left[Z'(\varphi)+2\frac{F'(\varphi)Z(\varphi)}{F(\varphi)}+5\frac{F'(\varphi)F''(\varphi)}{F(\varphi)}\right]D_i\varphi D^i\varphi-\left[2Z(\varphi)+3\frac{F'(\varphi)^2}{F(\varphi)}\right]D_i\ln\alpha D^i\varphi\\ &+\left[2Z(\varphi)+5\frac{F'(\varphi)^2}{F(\varphi)}\right]KK_\varphi+2\frac{F'(\varphi)}{F(\varphi)}\left[4\pi(S-3\rho)-3\kappa_1(1+\kappa_2)\Theta+2D_iZ^i-2\Theta K\right]\\ &+2U'(\varphi)-6\frac{F'(\varphi)}{F(\varphi)}U(\varphi)\Bigg\}.\end{aligned} \tag{B.2.47}$$

For evolving equation of K, we follow the same route we took in section B.1.2:

$$\begin{aligned}(\partial_t-\mathcal{L}_\beta)K &= (\partial_t-\mathcal{L}_\beta)[\gamma^{ij}K_{ij}]\\ &= K_{ij}(\partial_t-\mathcal{L}_\beta)\gamma^{ij}+\gamma^{ij}(\partial_t-\mathcal{L}_\beta)K_{ij}.\end{aligned} \tag{B.2.48}$$

Put equations (4.162) and (4.164) into it to get

$$\begin{aligned}\partial_t K = &-D^i D_i \alpha + \alpha\left[R + \frac{2}{F(\varphi)} D^i Z_i + K^2 - \frac{2}{F(\varphi)}\Theta K\right] + \beta^i \partial_i K - \frac{3}{F(\varphi)}\alpha\kappa_1(1+\kappa_2)\Theta \\ &- \frac{3}{F(\varphi)}\alpha U(\varphi) - \frac{5F''(\varphi)+2Z(\varphi)}{2F(\varphi)}\alpha D^i\varphi D_i\varphi - \frac{5F'(\varphi)}{2F(\varphi)}\alpha D^i D_i\varphi + \frac{5F'(\varphi)}{2F(\varphi)}\alpha K K_\varphi \\ &+ \frac{3F''(\varphi)}{2F(\varphi)}\alpha K_\varphi^2 - \frac{3F'(\varphi)}{2F(\varphi)} D^i\alpha D_i\varphi - \frac{3F'(\varphi)}{2F(\varphi)}\alpha \mathcal{L}_n K_\varphi + \frac{4\pi}{F(\varphi)}\alpha(S - 3\rho),\end{aligned} \tag{B.2.49}$$

where we have used

$$K_{ij}\left(\partial_t - \mathcal{L}_\beta\right)\gamma^{ij} = -K^{ij}\left(\partial_t - \mathcal{L}_\beta\right)\gamma_{ij}, \tag{B.2.50}$$

which is because

$$\begin{aligned}K_{ij}\partial_t\gamma^{ij} &= K^{mn}\gamma_{im}\gamma_{jn}\partial_t\gamma^{ij} \\ &= K^{mn}\left[\partial_t\left(\gamma_{im}\gamma_{jn}\gamma^{ij}\right) - \gamma_{jn}\gamma^{ij}\partial_t\left(\gamma_{im}\right) - \gamma_{im}\gamma^{ij}\partial_t\left(\gamma_{jn}\right)\right] \\ &= K^{mn}\partial_t\gamma_{mn} - K^{im}\partial_t\gamma_{im} - K^{jn}\partial_t\gamma_{jn} \\ &= -K^{ij}\partial_t\gamma_{ij},\end{aligned} \tag{B.2.51}$$

and

$$\begin{aligned}K_{ij}\mathcal{L}_\beta\gamma^{ij} &= K_{ij}\left[\beta^k\partial_k\gamma^{ij} - \gamma^{ik}\partial_k\beta^j - \gamma^{jk}\partial_k\beta^i\right] \\ &= -K^{ij}\beta^k\partial_k\gamma_{ij} - K^{mn}\gamma_{im}\gamma_{jn}\gamma^{ik}\partial_k\beta^j - K^{mn}\gamma_{im}\gamma_{jn}\gamma^{jk}\partial_k\beta^i \\ &= -K^{ij}\left[\beta^k\partial_k\gamma_{ij} + \gamma_{ik}\partial_j\beta^k + \gamma_{jk}\partial_i\beta^k\right] \\ &= -K^{ij}\mathcal{L}_\beta\gamma_{ij}.\end{aligned} \tag{B.2.52}$$

In the second equality of (B.2.52) we have used $K_{ij}\beta^k\partial_k\gamma^{ij} = -K^{ij}\beta^k\partial_k\gamma_{ij}$, which can be derived in the same way as Eq. (B.2.51), and in the third equality we have changed indices.

The evolving equation of $\tilde{A}_{ij}$ have a expression of

$$\begin{aligned}(\partial_t - \mathcal{L}_\beta)\tilde{A}_{ij} &= (\partial_t - \mathcal{L}_\beta)\phi^2(K_{ij} - \frac{1}{3}K\gamma_{ij}) \\ &= 2\phi(K_{ij} - \frac{1}{3}K\gamma_{ij})(\partial_t - \mathcal{L}_\beta)\phi + \phi^2(\partial_t - \mathcal{L}_\beta)K_{ij} \\ &\quad - \frac{1}{3}\phi^2\gamma_{ij}(\partial_t - \mathcal{L}_\beta)K - \frac{1}{3}\phi^2 K(\partial_t - \mathcal{L}_\beta)\gamma_{ij},\end{aligned} \tag{B.2.53}$$

Now use equations (4.61), (4.164), (B.2.49) and (4.162) and perform some intricate calculation, we get

$$\begin{aligned}\partial_t\tilde{A}_{ij} = \phi^2\Big[&- D_iD_j\alpha + \alpha R_{ij} + \frac{\alpha}{F(\varphi)}(D_iZ_j + D_jZ_i - 8\pi S_{ij})\Big]^{\mathrm{TF}} - \frac{2}{3}\tilde{A}_{ij}\partial_k\beta^k - 2\alpha\tilde{A}_{il}\tilde{A}^l_j \\ &+ \alpha\tilde{A}_{ij}\left[K - \frac{2\Theta}{F(\varphi)}\right] + 2\tilde{A}_{k(i}\partial_{j)}\beta^k + \beta^k\partial_k\tilde{A}_{ij} \\ &- \alpha\phi^2\left[\frac{F''(\varphi) + Z(\varphi)}{F(\varphi)}D_i\varphi D_j\varphi + \frac{F'(\varphi)}{F(\varphi)}D_iD_j\varphi\right]^{\mathrm{TF}} + \alpha\frac{F'(\varphi)}{F(\varphi)}K_\varphi\tilde{A}_{ij},\end{aligned} \tag{B.2.54}$$

where the TF index again means the trace is removed. In the above we have also use

$$\begin{aligned}\tilde{A}^l_i\tilde{A}_{jl} &= \tilde{\gamma}^{kl}\tilde{A}_{ik}\tilde{A}_{jl} \\ &= \phi^4\tilde{\gamma}^{kl}\left(K_{ik} - \frac{1}{3}K\gamma_{ik}\right)\left(K_{jl} - \frac{1}{3}K\gamma_{jl}\right) \\ &= \phi^2\gamma^{kl}\left(K_{ik} - \frac{1}{3}K\gamma_{ik}\right)\left(K_{jl} - \frac{1}{3}K\gamma_{jl}\right) \\ &= \phi^2K^l_iK_{jl} - \frac{2}{3}\phi^2KK_{ij} + \frac{1}{9}\phi^2K^2\gamma_{ij} \\ &= \phi^2K^l_iK_{jl} - \frac{1}{3}\phi^2KK_{ij} + \frac{1}{3}\phi^2K\tilde{A}_{ij}.\end{aligned} \tag{B.2.55}$$

The final equation we need to check is the one for $Gam\hat{m}a^{i}$, which now write as

$$\begin{aligned}\partial_t\hat{\Gamma}^i &= \partial_t(-\partial_j\tilde{\gamma}^{ij} + 2\tilde{\gamma}^{ij}Z_j) \\ &= -\partial_j\partial_t\tilde{\gamma}^{ij} + 2Z_j\partial_t\tilde{\gamma}^{ij} + 2\tilde{\gamma}^{ij}\partial_tZ_j.\end{aligned} \tag{B.2.56}$$

We then put the expression (4.162) and (4.167) into it. Notice the first and second term of the expression remain unchanged compared to its counterpart in standard gravity, and in the standard gravity part of the third term (derivatives of Z_i), the only quantities that are affected by the scalar field are

$$D_j K_i^j - D_i K \to F(\varphi)(D_j K_i^j - D_i K), \tag{B.2.57}$$

in the derivation, it would become

$$-\frac{4}{3}F(\varphi)\alpha\tilde{\gamma}^{ij}\partial_j K + 2F(\varphi)\alpha\tilde{\gamma}^{ij}\nabla_l \tilde{A}_j^l. \tag{B.2.58}$$

The second term above now replaces the second term in (B.1.19). Thus the first three terms in evolving equation of $\hat{\Gamma}^i$ is modified as:

$$\begin{aligned}2\alpha\bigg(\tilde{\Gamma}^i_{jk}\tilde{A}^{jk} - 3\tilde{A}^{ij}\frac{\partial_j\phi}{\phi} - \frac{2}{3}\tilde{\gamma}^{ij}\partial_j K\bigg) &\to \\ 2\alpha[F(\varphi)-1]\partial_j\tilde{A}^{ij} + 2\alpha F(\varphi)&\bigg(\tilde{\Gamma}^i_{jk}\tilde{A}^{jk} - 3\tilde{A}^{ij}\frac{\partial_j\phi}{\phi} - \frac{2}{3}\tilde{\gamma}^{ij}\partial_j K\bigg).\end{aligned} \tag{B.2.59}$$

The extra terms due to φ are

$$\begin{aligned}&2\tilde{\gamma}^{ij}\alpha\bigg[-[Z(\varphi)+F''(\varphi)]K_\varphi D_j\varphi - F'(\varphi)D_j K_\varphi + F'(\varphi)K_j^k D_k\varphi\bigg] = \\ &\quad 2\tilde{\gamma}^{ij}\bigg[-\alpha[Z(\varphi)+F''(\varphi)]K_\varphi D_j\varphi - \alpha F'(\varphi)D_j K_\varphi + \alpha F'(\varphi)\tilde{A}_j^k D_k\varphi + \frac{1}{3}\alpha F'(\varphi)K D_j\varphi\bigg]\end{aligned} \tag{B.2.60}$$

We have used the relation

$$\tilde{A}_j^k = \phi^{-2}\gamma^{kl}[\phi^2(K_{lj} - \frac{1}{3}K\gamma_{lj})] = K_j^k - \frac{1}{3}K\gamma_j^k \to K_j^k = \tilde{A}_j^k + \frac{1}{3}K\delta_j^k. \tag{B.2.61}$$

The final equation is given as

$$
\begin{aligned}
\partial_t\hat{\Gamma}^i = {} & 2\alpha[F(\varphi)-1]\partial_j\tilde{A}^{ij} + 2\alpha F(\varphi)\left(\tilde{\Gamma}^i_{jk}\tilde{A}^{jk} - 3\tilde{A}^{ij}\frac{\partial_j\phi}{\phi} - \frac{2}{3}\tilde{\gamma}^{ij}\partial_j K\right) \\
& + 2\tilde{\gamma}^{ki}\left(\alpha\partial_k\Theta - \Theta\partial_k\alpha - \frac{2}{3}\alpha K Z_k\right) - 2\tilde{A}^{ij}\partial_j\alpha + \tilde{\gamma}^{kl}\partial_k\partial_l\beta^i + \frac{1}{3}\tilde{\gamma}^{ik}\partial_k\partial_l\beta^l + \frac{2}{3}\tilde{\Gamma}^i\partial_k\beta^k \\
& - \tilde{\Gamma}^k\partial_k\beta^i + 2\kappa_3\left(\frac{2}{3}\tilde{\gamma}^{ij}Z_j\partial_k\beta^k - \tilde{\gamma}^{jk}Z_j\partial_k\beta^i\right) + \beta^k\partial_k\hat{\Gamma}^i - 2\alpha\kappa_1\tilde{\gamma}^{ij}Z_j - 16\pi\alpha\tilde{\gamma}^{ij}S_j \\
& + 2\tilde{\gamma}^{ij}\Big[-\alpha[Z(\varphi)+F''(\varphi)]K_\varphi D_j\varphi - \alpha F'(\varphi)D_jK_\varphi + \alpha F'(\varphi)\tilde{A}^k_j D_k\varphi + \frac{1}{3}\alpha F'(\varphi)KD_j\varphi\Big].
\end{aligned}
\tag{B.2.62}
$$

The introduction of auxiliary variables (4.68) and (4.168) in the evolving equations for the scalar field gravity is completely the same as we did for standard gravity in section B.1.2, e.g., for the new auxiliary variable Π_i, we have

$$
\begin{gathered}
\partial_t\Pi_i = \partial_t\partial_i\varphi = \partial_i\partial_t\varphi = \partial_i(-\alpha K_\varphi + \beta^k\partial_k\varphi), \\
\Downarrow \\
\partial_t\Pi_i - \beta^k\partial_k\Pi_i + \alpha\partial_i K_\varphi = -\alpha A_i K_\varphi + B_i^k\Pi_k.
\end{gathered}
\tag{B.2.63}
$$

We do not cover the rest equations in detail here. Please refer to the (4.174) - (4.184) for the final evolving system of FOCCZ4 under the scalar field gravity.

APPENDIX C

Detailed Setup of the simple wave test

In the appendix, we explain how we construct the initial condition of the simple wave scenarios in our numerical relativity application. For more comprehensive information on these scenarios and their usage in testing numerical relativity code, one can refer to the publication (e.g., Alcubierre et al., 2003b).

For clarity, we provide the full list of evolving variables of our FOCCZ4 system here as a reference:

$[0]\tilde{\gamma}_{xx}, [1]\tilde{\gamma}_{xy}, [2]\tilde{\gamma}_{xz}, [3]\tilde{\gamma}_{yy}, [4]\tilde{\gamma}_{yz}, [5]\tilde{\gamma}_{zz},$

$[6]\tilde{A}_{xx}, [7]\tilde{A}_{xy}, [8]\tilde{A}_{xz}, [9]\tilde{A}_{yy}, [10]\tilde{A}_{yz}, [11]\tilde{A}_{zz},$

$[12]\Theta, [13]\hat{\Gamma}^x, [14]\hat{\Gamma}^y, [15]\hat{\Gamma}^z,$

$[16]\alpha, [17]\beta^x, [18]\beta^y, [19]\beta^z, [20]b^x, [21]b^y, [22]b^z,$

$[23]A_x, [24]A_y, [25]A_z,$

$[26]B_x^x, [27]B_x^y, [28]B_x^z, [29]B_y^x, [30]B_y^y, [31]B_y^z, [32]B_z^x, [33]B_z^y, [34]B_z^z,$

$[35]D_{xxx}, [36]D_{xxy}, [37]D_{xxz}, [38]D_{xyy}, [39]D_{xyz}, [40]D_{xzz},$

$[41]D_{yxx}, [42]D_{yxy}, [43]D_{yxz}, [44]D_{yyy}, [45]D_{yyz}, [46]D_{yzz},$

$[47]D_{zxx}, [48]D_{zxy}, [49]D_{zxz}, [50]D_{zyy}, [51]D_{zyz}, [52]D_{zzz},$

$[53]K, [54]\phi, [55]P_x, [56]P_y, [57]P_z.$

C.1 Linear wave test

This test examined the behaviour of a linear (gravitational) wave propagating through the domain. The magnitude of the wave needs to be relatively small, which allows the propagation to remain within the linear regime. The linearly helps to simplify the system significantly. The spacetime metric of this scenario is given as

$$ds^2 = -dt^2 + dx^2 + (1+H)dy^2 + (1-H)dz^2, \quad H(x,t) = \varepsilon \sin[k\pi(x-t)], \tag{C.1.1}$$

where $\varepsilon \to 0$ to make sure the wave stay linearized. From this, we get

$$\alpha = 1,\ \beta^i = 0, \phi = 1. \tag{C.1.2}$$

$$K_{yy} = -\frac{\partial_t H}{2\alpha} = -\frac{1}{2}\partial_t H,\ K_{zz} = -\frac{\partial_t H}{2\alpha} = \frac{1}{2}\partial_t H,\ K_{ij,\text{others}} = 0. \tag{C.1.3}$$

$$K = \gamma^{ij}K_{ij} = -\frac{1}{2}(1-H)\partial_t H + \frac{1}{2}(1+H)\partial_t H = 0. \tag{C.1.4}$$

We then assign the evolving variables accordingly:

$$Q[0] = \tilde{\gamma}_{xx} = 1, \tag{C.1.5}$$

$$Q[3] = \tilde{\gamma}_{yy} = 1 + \varepsilon \sin[k\pi(x-t)], \tag{C.1.6}$$

$$Q[5] = \tilde{\gamma}_{zz} = 1 - \varepsilon \sin[k\pi(x-t)], \tag{C.1.7}$$

$$Q[9] = \tilde{A}_{yy} = (K_{yy} - \frac{1}{3}\gamma_{yy}K) = K_{yy} = \frac{1}{2}\varepsilon k\pi \cos[k\pi(x-t)], \tag{C.1.8}$$

$$Q[11] = \tilde{A}_{zz} = (K_{zz} - \frac{1}{3}\gamma_{yy}K) = K_{zz} = -\frac{1}{2}\varepsilon k\pi \cos[k\pi(x-t)], \tag{C.1.9}$$

$$Q[16] = \alpha = 0, \tag{C.1.10}$$

$$Q[38] = D_{xyy} = \frac{1}{2}\partial_x\tilde{\gamma}_{yy} = \frac{1}{2}\varepsilon k\pi \cos[k\pi(x-t)], \tag{C.1.11}$$

$$Q[40] = D_{xzz} = \frac{1}{2}\partial_x\tilde{\gamma}_{zz} = -\frac{1}{2}\varepsilon k\pi \cos[k\pi(x-t)], \tag{C.1.12}$$

$$Q[54] = \phi = 0. \tag{C.1.13}$$

Other quantities that do not mention above are all 0. In code practices, we usually set the wave magnitude to be under 10^{-4}. Please notice the t variables are kept in the formulation only for consistency and We set $t = 0$ when we assign the initial conditions in the code. Those expressions are not used in the stage of evolution. The same principle also applies to the gauge wave scenario below.

C.2 Gauge wave test

The gauge wave scenario derives its name as being a time-dependent coordinate transformation applied to the flat Minkowski spacetime. No physics phenomenon occurs here, but as we dynamically slice the spacetime, the system becomes time-dependent which is quite suitable for testing numerical relativity code.

C.2.1 1D Gauge Wave Setup

The spacetime metric of the 1D gauge wave setup is

$$ds^2 = -H(x,t)dt^2 + H(x,t)dx^2 + dy^2 + dz^2, \;\; H(x,t) = 1 - A\sin[k\pi(x-t)]. \tag{C.2.14}$$

from the metric we get

$$\alpha = \sqrt{H}, \; \beta^i = 0, \; \phi = H^{-1/6}, \tag{C.2.15}$$

and

$$K_{xx} = -\frac{\partial_t H}{2\alpha} = -\frac{k\pi A}{2}\frac{\cos[k\pi(x-t)]}{\{1 - A\sin[k\pi(x-t)]\}^{1/2}}, \quad K_{ij,\text{others}} = 0, \tag{C.2.16}$$

$$K = \gamma^{ij}K_{ij} = \frac{K_{xx}}{H} = -\frac{k\pi A}{2}\frac{\cos[k\pi(x-t)]}{\{1 - A\sin[k\pi(x-t)]\}^{3/2}}. \tag{C.2.17}$$

We then give the full list of initial quantities of the system accordingly:

$$Q[0] = \tilde{\gamma}_{xx} = \phi^2 H = \{1 - A\sin[k\pi(x-t)]\}^{2/3}, \tag{C.2.18}$$

$$Q[3] = \tilde{\gamma}_{yy} = \phi^2 = \{1 - A\sin[k\pi(x-t)]\}^{-1/3}, \tag{C.2.19}$$

$$Q[5] = \tilde{\gamma}_{zz} = \phi^2 = \{1 - A\sin[k\pi(x-t)]\}^{-1/3}, \tag{C.2.20}$$

$$Q[6] = \tilde{A}_{xx} = \phi^2(K_{xx} - \frac{1}{3}\gamma_{xx}K) = -\frac{k\pi A}{3}\frac{\cos[k\pi(x-t)]}{\{1 - A\sin[k\pi(x-t)]\}^{5/6}}, \tag{C.2.21}$$

$$Q[9] = \tilde{A}_{yy} = \phi^2(0 - \frac{1}{3}\gamma_{yy}K) = \frac{k\pi A}{6}\frac{\cos[k\pi(x-t)]}{\{1 - A\sin[k\pi(x-t)]\}^{11/6}}, \tag{C.2.22}$$

$$Q[11] = \tilde{A}_{zz} = \phi^2(0 - \frac{1}{3}\gamma_{zz}K) = \frac{k\pi A}{6}\frac{\cos[k\pi(x-t)]}{\{1 - A\sin[k\pi(x-t)]\}^{11/6}}, \tag{C.2.23}$$

$$Q[13] = \hat{\Gamma}^x = \tilde{\Gamma}^x = -\partial_j\tilde{\gamma}^{jx} = \frac{2}{3}H^{-5/3}\partial_x H = -\frac{2k\pi A}{3}\frac{\cos[k\pi(x-t)]}{\{1 - A\sin[k\pi(x-t)]\}^{5/3}}, \tag{C.2.24}$$

$$Q[16] = \alpha = \sqrt{H} = \sqrt{1 - A\sin[k\pi(x-t)]}, \tag{C.2.25}$$

$$Q[23] = A_x = \partial_x\alpha = \frac{\partial_x H}{2\sqrt{H}} = -\frac{k\pi A}{2}\frac{\cos[k\pi(x-t)]}{\sqrt{1 - A\sin[k\pi(x-t)]}}, \tag{C.2.26}$$

$$Q[35] = D_{xxx} = \frac{1}{2}\partial_x(\phi^2 H) = \frac{1}{3}H^{-1/3}\partial_x H = -\frac{k\pi A}{3}\frac{\cos[k\pi(x-t)]}{\{1 - A\sin[k\pi(x-t)]\}^{1/3}}, \tag{C.2.27}$$

$$Q[38] = D_{xyy} = \frac{1}{2}\partial_x\phi^2 = -\frac{1}{6}H^{-4/3}\partial_x H = \frac{k\pi A}{6}\frac{\cos[k\pi(x-t)]}{\{1 - A\sin[k\pi(x-t)]\}^{4/3}}, \tag{C.2.28}$$

$$Q[40] = D_{xzz} = \frac{1}{2}\partial_x\phi^2 = -\frac{1}{6}H^{-4/3}\partial_x H = \frac{k\pi A}{6}\frac{\cos[k\pi(x-t)]}{\{1 - A\sin[k\pi(x-t)]\}^{4/3}}, \tag{C.2.29}$$

$$Q[53] = K = -\frac{k\pi A}{2}\frac{\cos[k\pi(x-t)]}{\{1 - A\sin[k\pi(x-t)]\}^{3/2}}, \tag{C.2.30}$$

$$Q[54] = \phi = \sqrt{1 - A\sin[k\pi(x-t)]}^{-1/6}, \tag{C.2.31}$$

$$Q[55] = \partial_x\phi = -\frac{1}{6}H^{-7/6}\partial_x H = \frac{k\pi A}{6}\frac{\cos[k\pi(x-t)]}{\{1 - A\sin[k\pi(x-t)]\}^{7/6}}. \tag{C.2.32}$$

Again, other quantities that are not mentioned above are all 0. The common choice of wave magnitude for the gauge wave is 0.1 in our tests.

C.2.2 2D (diagonal) Gauge Wave Setup

The metric of a (2D) gauge wave can be calculated from a coordinate transformation of its 1D counterpart:

$$dt' = dt,\ dx' = \frac{1}{\sqrt{2}}(x-y),\ dy' = \frac{1}{\sqrt{2}}(x+y),\ dz' = dz. \tag{C.2.33}$$

Below we will continue to use quantities without prime for convenience. Assume now our H function takes a form of

$$H(x, y, t) = 1 + 2B \sin[k\pi(x - y - t)]. \tag{C.2.34}$$

Two things get changed here compared to the 1D case: the perturbing term is now positive, and the wave velocity is reduced (i.e., no $\sqrt{2}$ before t) for simplicity. We then calculate the new line element for the spacetime:

$$\begin{aligned} ds^2 = &[1 + B \sin[k\pi(x - y - t)]]\, dx^2 + [1 + B \sin[k\pi(x - y - t)]]\, dy^2 \\ &- 2B \sin[k\pi(x - y - t)] dxdy - H dt^2 + dz^2. \end{aligned} \tag{C.2.35}$$

For clarity, we write down its space metric and inverse one:

$$\gamma_{ij} = \begin{pmatrix} 1 + B\sin\Omega & -B\sin\Omega & 0 \\ -B\sin\Omega & 1 + B\sin\Omega & 0 \\ 0 & 0 & 1 \end{pmatrix}, \gamma^{ij} = \begin{pmatrix} \frac{1}{H}(1 + B\sin\Omega) & \frac{1}{H} B\sin\Omega & 0 \\ \frac{1}{H} B\sin\Omega & \frac{1}{H}(1 + B\sin\Omega) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{C.2.36}$$

and the conformal metric and its inverse:

$$\tilde{\gamma}_{ij} = H^{-1/3} \begin{pmatrix} 1 + B\sin\Omega & -B\sin\Omega & 0 \\ -B\sin\Omega & 1 + B\sin\Omega & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{C.2.37}$$

$$\tilde{\gamma}^{ij} = H^{1/3} \begin{pmatrix} \frac{1}{H}(1 + B\sin\Omega) & \frac{1}{H} B\sin\Omega & 0 \\ \frac{1}{H} B\sin\Omega & \frac{1}{H}(1 + B\sin\Omega) & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{C.2.38}$$

We have defined $\Omega \equiv k\pi(x - y - t)$ for simplicity. From the line element, we can still read

$$\alpha = \sqrt{H},\ \beta^i = 0,\ \phi = H^{-1/6}, \tag{C.2.39}$$

and (use $\partial_t \gamma_{ij} = -2\alpha K_{ij}$):

$$K_{ij} = \begin{pmatrix} \frac{k\pi B}{2\sqrt{H}}\cos\Omega & -\frac{k\pi B}{2\sqrt{H}}\cos\Omega & 0 \\ -\frac{k\pi B}{2\sqrt{H}}\cos\Omega & \frac{k\pi B}{2\sqrt{H}}\cos\Omega & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad K = \gamma^{ij} K_{ij} = \frac{k\pi B}{H^{3/2}}\cos\Omega. \tag{C.2.40}$$

The full initial condition is thus given as:

$$Q[0] = \tilde{\gamma}_{xx} = H^{-1/3}(1 + B\sin\Omega), \tag{C.2.41}$$

$$Q[1] = \tilde{\gamma}_{xy} = -H^{-1/3} B\sin\Omega, \tag{C.2.42}$$

$$Q[3] = \tilde{\gamma}_{yy} = H^{-1/3}(1 + B\sin\Omega), \tag{C.2.43}$$

$$Q[5] = \tilde{\gamma}_{zz} = H^{-1/3}, \tag{C.2.44}$$

$$Q[6] = \tilde{A}_{xx} = \phi^2\left(K_{xx} - \frac{1}{3}\gamma_{xx}K\right) = \frac{k\pi B}{2H^{5/6}}\cos\Omega\left[1 - \frac{2}{3H}(1 + B\sin\Omega)\right], \tag{C.2.45}$$

$$Q[7] = \tilde{A}_{xy} = \phi^2\left(K_{xy} - \frac{1}{3}\gamma_{xy}K\right) = \frac{k\pi B}{2H^{5/6}}\cos\Omega\left[\frac{2}{3H}B\sin\Omega - 1\right], \tag{C.2.46}$$

$$Q[9] = \tilde{A}_{yy} = \tilde{A}_{xx}, \tag{C.2.47}$$

$$Q[11] = \tilde{A}_{zz} = \phi^2\left(0 - \frac{1}{3}\gamma_{zz}K\right) = \frac{k\pi B}{3H^{11/6}}\cos\Omega, \tag{C.2.48}$$

$$Q[13] = \hat{\Gamma}^x = \tilde{\Gamma}^x = -\partial_j \tilde{\gamma}^{jx} = -\partial_x \tilde{\gamma}^{xx} - \partial_y \tilde{\gamma}^{xy} \tag{C.2.49}$$

$$= \frac{k\pi B}{3H^{5/3}}\left[(1 - 2B\sin\Omega) + (3 + 2B\sin\Omega)\right]\cos\Omega = \frac{4k\pi B}{3H^{5/3}}\cos\Omega, \tag{C.2.50}$$

$$Q[14] = \hat{\Gamma}^y = \tilde{\Gamma}^y = -\partial_j \tilde{\gamma}^{jy} = -\partial_x \tilde{\gamma}^{xy} - \partial_y \tilde{\gamma}^{yy} = -\frac{4k\pi B}{3H^{5/3}}\cos\Omega, \tag{C.2.51}$$

$$Q[16] = \alpha = \sqrt{H}, \tag{C.2.52}$$

$$Q[23] = A_x = \partial_x \alpha = \frac{\partial_x H}{2\sqrt{H}} = \frac{k\pi B}{\sqrt{H}}\cos\Omega, \tag{C.2.53}$$

$$Q[24] = A_y = \partial_y \alpha = \frac{\partial_y H}{2\sqrt{H}} = -\frac{k\pi B}{\sqrt{H}}\cos\Omega, \tag{C.2.54}$$

$$Q[35] = D_{xxx} = \frac{1}{2}\partial_x \tilde{\gamma}_{xx} = \frac{k\pi B}{6H^{4/3}}\cos\Omega(1 + 4B\sin\Omega), \tag{C.2.55}$$

$$Q[36] = D_{xxy} = \frac{1}{2}\partial_x \tilde{\gamma}_{xy} = -\frac{k\pi B}{6H^{4/3}}\cos\Omega(3 + 4B\sin\Omega), \tag{C.2.56}$$

$$Q[38] = D_{xyy} = \frac{1}{2}\partial_x \tilde{\gamma}_{yy} = D_{xxx}, \tag{C.2.57}$$

$$Q[40] = D_{xzz} = \frac{1}{2}\partial_x \tilde{\gamma}_{zz} = -\frac{k\pi B}{3H^{4/3}}\cos\Omega, \tag{C.2.58}$$

$$Q[41] = D_{yxx} = \frac{1}{2}\partial_y \tilde{\gamma}_{xx} = -\frac{k\pi B}{6H^{4/3}}\cos\Omega(1 + 4B\sin\Omega), \tag{C.2.59}$$

$$Q[42] = D_{yxy} = \frac{1}{2}\partial_y \tilde{\gamma}_{xy} = \frac{k\pi B}{6H^{4/3}} \cos\Omega(3 + 4B\sin\Omega), \tag{C.2.60}$$

$$Q[44] = D_{yyy} = \frac{1}{2}\partial_y \tilde{\gamma}_{yy} = D_{yxx}, \tag{C.2.61}$$

$$Q[46] = D_{yzz} = \frac{1}{2}\partial_y \tilde{\gamma}_{zz} = \frac{k\pi B}{3H^{4/3}} \cos\Omega, \tag{C.2.62}$$

$$Q[53] = K = \frac{k\pi B}{H^{3/2}} \cos\Omega, \tag{C.2.63}$$

$$Q[54] = \phi = H^{-1/6}, \tag{C.2.64}$$

$$Q[55] = P_x = \partial_x \phi = -\frac{1}{6}H^{-7/6}\partial_x H = -\frac{k\pi B}{3H^{7/6}} \cos\Omega, \tag{C.2.65}$$

$$Q[56] = P_y = \partial_y \phi = -\frac{1}{6}H^{-7/6}\partial_y H = \frac{k\pi B}{3H^{7/6}} \cos\Omega. \tag{C.2.66}$$

Again, other quantities that are not mentioned above are all 0.