IMPLEMENTING A MESHLESS FINITE MASS SCHEME IN THE COSMOLOGICAL N-BODY CODE GADGET

Master's Thesis

by

Paul Marten Hinz

born 29 December 1994 in Greifswald, Germany

Supervised by

PD Dr. Klaus Dolag

Universitäts-Sternwarte München, Germany

&

Dr. Rhea-Silvia Remus

Universitäts-Sternwarte München, Germany

2020

Ludwig-Maximilians-Universität München

IMPLEMENTIERUNG EINES MESHLESS-FINITE-MASS-SCHEMAS IN DEN KOSMOLOGISCHEN N-KÖRPER-CODE GADGET

Masterarbeit

von

Paul Marten Hinz

geboren 29. Dezember 1994 in Greifswald

Unter Aufsicht von

PD Dr. Klaus Dolag

Universitäts-Sternwarte München

&

Dr. Rhea-Silvia Remus

Universitäts-Sternwarte München

2020

Ludwig-Maximilians-Universität München

Abstract

We implemented a meshless finite mass (MFM) scheme to resolve fluid dynamics in the cosmological N-body code `OpenGadget3`. In doing so, we offer an alternative method of simulating hydrodynamics in astrophysical environments, contrasted to the historic approach of smoothed particle hydrodynamics (SPH). The new scheme directly calculates inter-particle fluxes while still maintaining an overall Lagrangian nature. We discuss the theory behind both solvers and carry out various test cases to study their performances. We find that for simulations with exclusively hydrodynamical interactions, MFM produces less over-smoothing while executing in roughly half the time of SPH. This comes with the drawback of noisier solutions. Coupling our implementation to gravity, we see good agreement between SPH and MFM for simple tests but find numerical and physical instabilities for more demanding cases. We believe these to be resolvable without requiring a complete rewrite of the solver and expect MFM to become a viable competitor to SPH in the future.

Zusammenfassung

In den kosmologischen N-Körper-Code `OpenGadget3` wurde ein Meshless-Finite-Mass-Algorithmus (MFM) eingebaut, um Fluiddynamiken zu simulieren. Wir bieten somit eine alternative Methode zum historischen Ansatz Smoothed Particle Hydrodynamics (SPH) an, Hydrodynamik in astrophysikalischen Systemen numerisch aufzulösen. Unser neues Schema berechnet direkte Flüsse zwischen Teilchen, während der grundsätzliche Lagrange-Ansatz gewahrt wird. Wir stellen die Theorie hinter beiden Algorithmen vor und studieren ihre Performance anhand zahlreicher Tests. Für Simulationen rein hydrodynamischer Natur zeigt sich, dass MFM physikalische Größen weniger ausschmiert als SPH und zugleich nur halb so viel Zeit für die Auswertung benötigt. Dies birgt den Nachteil von erhöhtem Rauschen der Lösungen. Bei Einbindung von Gravitation in die Simulationen vermerken wir eine gute Übereinstimmung beider Ansätze für einfache Tests. Komplexere Probleme hingegen offenbaren numerische und physikalische Instabilitäten unserer MFM-Implementierung. Wir nehmen an, dass diese behebbar sind, ohne den Algorithmus von Grund auf neu zu schreiben und erwarten, dass sich MFM in Zukunft als veritable Konkurrenz zu SPH etabliert.

Contents

1 Motivation

In astronomy, more so than in other sciences, our abilities to observe the very objects we set out to study are quite limited. With a few exceptions (such as neutrinos, see e.g. Nobel Prize laureates of 1995, 2002, and 2015; or the newly found gravitational waves, Nobel Prize of 2017), we are limited to the detection of light. But the farther out the objects of our interest lie, the dimmer the light and the less information we are able to deduce. It seems a daunting task to study the most distant regions in our Observable Universe using only a handful of photons. With the soaring development of computers over the past several decades it is thus only natural that astronomical simulations have become a popular means to study our cosmos and the objects therein in much more detail than we would be able to using observations alone.

This approach, however, brings about some inherent difficulties as well. Discretizing the continuous laws of nature must be done with the highest of caution and while computational power is now available more abundantly than ever before, it is still finite and, on large scales, expensive. When setting up simulations, our two main goals therefore always ought to be that they run as precisely and as effectively as possible.

This Master's thesis is focussed on a distinct example that highlights how these two goals can be pursued. In an established astrophysical code, a meshless finite mass (MFM) scheme, an alternative method to solve the hydrodynamical equations for gas interactions, was implemented, aiming at making the existing code both execute faster and produce more reliable results at the same time.

First, we will start by introducing the fundamentals of a numerical treatment of fluid dynamics in Sec. 2. Here, we will also introduce the theory of smoothed particle hydrodynamics (SPH), the predecessor to our MFM implementation to which we will draw ample comparisons later. In Sec. 3, the theory behind meshless schemes in general and MFM in particular shall be discussed. We will then introduce the code that we were working with, `GADGET`, in Sec. 4 before Sec. 5 focuses on the implementation of an MFM scheme therein. We will highlight different aspects of said implementation using a variety of well-understood test cases and evaluate its performance. We will continue by discussing our results in Sec. 6 and lastly, Sec. 7 will feature a conclusive summary as well as an outlook for future endeavours.

2 Introduction to Computational Fluid Dynamics

The general goal of computational fluid dynamics (CFD) is quite easily stated: to accurately reproduce a realistic evolution of a fluid (i.e. a liquid or gas) using numerical methods. To achieve this goal, many methods have been introduced since the advent of computers as a scientific tool, each of which with its own merits, difficulties, and pitfalls. This section shall present the equations governing the nature of fluid dynamics and the two most common approaches of implementing CFD in astrophysics[1].

2.1 The Equations of Fluid Dynamics

Following Landau and Lifshitz (2013), we start our theoretical examination by considering a fluid on macroscopic scales. Its state is completely determined by its velocity field $\vec{v}(x, y, z, t)$ and any two thermodynamic quantities such as its density and pressure fields, $\rho(x, y, z, t)$ and $p(x, y, z, t)$, respectively, together with an equation of state relating these to other thermodynamic variables. Note that all of the above quantities posses an explicit time dependence. Our task now is to establish how this evolution with time can be quantified.

In the following, we will assume our fluid to be ideal, i.e. that thermal conductivity and viscosity can be neglected. The first of the two assumptions also imposes another constraint, namely that all motions must happen adiabatically:

$$\frac{\mathrm{d}s}{\mathrm{d}t} = 0 \tag{2.1}$$

with the entropy per unit mass s.

Continuity equation We consider an arbitrary (not necessarily infinitesimal) volume element of the fluid and denote it by V^*. Naturally, the total mass contained within it is equal to $\int \rho \, \mathrm{d}V$, evaluated over all of V^*. The mass flow into V^* can now trivially be identified with $-\frac{\partial}{\partial t} \int \rho \, \mathrm{d}V$. We put this result aside and now consider the mass flow out of V^* through a surface element $\mathrm{d}\vec{A}$. Choosing the direction of $\mathrm{d}\vec{A}$ to be along the normal pointing outwards, the mass flow can be expressed as $\rho\vec{v}\,\mathrm{d}\vec{A}$ with positive (negative) values indicating a net outflow (inflow). Integrating over the whole surface of the volume, we find the total mass flow to be equal to $\oint \rho\vec{v}\,\mathrm{d}\vec{A}$. We can now equate this with our first result (keeping the signs in mind) to obtain

$$-\frac{\partial}{\partial t} \int \rho \, \mathrm{d}V = \oint \rho\vec{v}\,\mathrm{d}\vec{A} \; . \tag{2.2}$$

Using Gauss's theorem to rewrite the surface integral to a volume integral, bringing both terms to one side, and then using Leibniz's rule to interchange the integral and differential operators, we find

$$\int \frac{\partial \rho}{\partial t} + \nabla(\rho\vec{v}) \, \mathrm{d}V = 0 \; . \tag{2.3}$$

We can now use the universality of Eq. (2.3) to conclude that the integrand must vanish in all cases and thus find the equation of continuity:

$$\frac{\partial \rho}{\partial t} + \nabla(\rho\vec{v}) = \frac{\partial \rho}{\partial t} + \rho\nabla\vec{v} + \vec{v}\,\nabla\rho = 0 \; . \tag{2.4}$$

[1]The applications of CFD in astrophysical contexts cover a wide range of scales, from resolving the gas contents of galaxies and clusters (as is the case in `GADGET`) to the simulation of stellar interiors.

Euler's equation Once again, the start of our argumentation shall be a finite volume element V^*. The total force acting on V^* is $-\oint p \, \mathrm{d}\vec{A}$ or, expressed as a volume integral, $-\int \nabla p \, \mathrm{d}V$. The integrand of the second expression can be understood as the force acting on any infinitesimal volume element which we can equate with Newton's second law as

$$-\nabla p = \rho \frac{\mathrm{d}\vec{v}}{\mathrm{d}t} \; . \tag{2.5}$$

Note that the time differentiation in Eq. (2.5) is a total one that expands as

$$\frac{\mathrm{d}\vec{v}}{\mathrm{d}t} = \frac{\partial \vec{v}}{\partial t} + (\vec{v}\,\nabla)\,\vec{v} \; . \tag{2.6}$$

Combining Eqs. (2.5) and (2.6), we find Euler's equation as

$$\frac{\partial \vec{v}}{\partial t} + (\vec{v}\,\nabla)\,\vec{v} = -\frac{1}{\rho}\nabla p \; . \tag{2.7}$$

Energy equation One of our core assumptions was to regard thermal conductivity as negligible. We can now make use of this to write the first law of thermodynamics in a simplified form:

$$\mathrm{d}U = -p \, \mathrm{d}V \tag{2.8}$$

with the internal energy U. We can find another expression for $\mathrm{d}V$ when considering the relative change of a volume element V with time. This is given as

$$\frac{1}{V}\frac{\mathrm{d}V}{\mathrm{d}t} = \nabla \vec{v} \; . \tag{2.9}$$

Combining Eqs. (2.8) and (2.9) as well as substituting U for the specific internal energy $u = U/(\rho V)$ we obtain an equation quantifying the energy evolution of our fluid:

$$\rho \frac{\mathrm{d}u}{\mathrm{d}t} = -p \nabla \vec{v} \; . \tag{2.10}$$

With Eqs. (2.4), (2.7), and (2.10), we now have enough information to accurately describe the evolution of our fluid indefinitely. The only other constraint we have to impose is an equation of state. A natural choice is the one for an ideal gas:

$$p = (\gamma - 1)\rho u \tag{2.11}$$

with the adiabatic index γ.

To solve these equations consistently using numerical methods, one first has to choose a discretization scheme. After all, a fully analytical treatment will not be possible in any case; discretization is the only way that programs will yield plausible results in finite time. The two most popular approaches discretize either mass (so-called Lagrangian methods) or volume (Eulerian methods). Both of them shall be introduced in the two following sections.

To end this general discussion, a word of caution ought to be stated. The general assumptions we made (no thermal conduction, no viscosity) are reasonable in quiet environments, especially when keeping in mind that each discretization element will represent a multitude of astrophysical objects bundled together. In more violent regions (e.g. in shock environments), however, these assumptions quickly become erroneous. To ensure that results stay as accurate as possible nonetheless, one must either solve the more general Navier-Stokes equations (that shall not be discussed further here) or introduce correction functions.

2.2 Lagrangian Methods: Smoothed Particle Hydrodynamics

In Lagrangian hydrodynamical codes, the fluid is represented as an ensemble of particles. This assumption alone makes Lagrangian codes very suitable to be combined with so-called N-body codes who use a similar ansatz to simulate gravity. It should be noted that in the majority of applications, the term *particle* does not in fact denote a singular physical particle but rather a merged subsample of all particles within the fluid to be described. How many physical particles comprise a numerical particle depends both on the scale of the simulation and its resolution. In cosmological codes, a numerical particle will often hold a total mass in excess of $10^3\,\mathrm{M}_\odot$. In the remainder of this work, *particle* shall always refer to a numerical particle as described here.

The exact implementation of Lagrangian methods can be done following various different approaches and philosophies. In this section, we will focus on the method of smoothed particle hydrodynamics (SPH, first implementations by Gingold and Monaghan 1977 and Lucy 1977), a popular candidate in astrophysical CFD and `GADGET`'s historic hydrosolver of choice. We will do so by following the excellent review paper by Price (2012) to which we refer the interested reader for much more details than fits the scope of this thesis. In Sec. 3, we will then introduce in detail so-called meshless methods, the very topic of this work. We also note, however, that there are other methods available, even though they are not as commonly used in astrophysics, such as particle-in-cell schemes (see e.g. Park et al. 2013 and Lee et al. 2019) or Osiptsov's method (see Lebedeva and Osiptsov 2016 for its theory and Mishchenko et al. 2020 for a recent application).

2.2.1 The Interpolation Problem and Its Solution in SPH

While the choice of a Lagrangian over an Eulerian scheme offers some intriguing advantages, such as the aforementioned facilitated coupling to others physics modules like gravity, it imposes one very striking problem: How can one describe continuous fluid properties using a discrete and finite set of particles? Or, to apply this problem to a concrete example: How can one calculate densities, given only a set of point masses at known positions? The idea at the very core of SPH schemes is to do so by using geometrically weighted averaging:

$$\rho(\vec{r}_0) = \sum_i^{N_\mathrm{ngb}} m_i \; W\left(\left|\vec{r}_0 - \vec{r}_i\right|, h\right) \; . \tag{2.12}$$

N_ngb is the number of neighbouring particles to an arbitrary position $\vec{r}_0$, and m_i and $\vec{r}_i$ are the mass and position, respectively, of each summed over neighbour particle i. The intricacies of SPH lie in the two remaining quantities, the kernel function W and the smoothing length h that shall be discussed in more detail in Sec. 2.2.3. For now, suffice it to say that W is a monotonically decreasing, radially symmetric function that weighs the contributions of particles closer to r_0 more than those of particles further away, and h is a parameter we can use to fine-tune the behaviour of W.

Having calculated the density at $\vec{r}_0$, we can immediately use this to calculate any other arbitrary quantity $X(\vec{r}_0)$ (both of scalar and vectorial nature) by realizing that

$$\lim_{h\to 0} W\left(\left|\vec{r}_0 - \vec{r}_i\right|, h\right) = \delta\left(\left|\vec{r}_0 - \vec{r}_i\right|\right) \tag{2.13}$$

with δ indicating Dirac's delta function. It thus follows that

$$\begin{aligned} X(\vec{r}_0) &= \int X(\vec{r})\ \delta\left(|\vec{r}_0 - \vec{r}|\right) \mathrm{d}\vec{r} \\ &\approx \int \frac{X(\vec{r})}{\rho(\vec{r})}\ W\left(|\vec{r}_0 - \vec{r}|, h\right) \rho(\vec{r})\, \mathrm{d}\vec{r} \\ &\approx \sum_i^{N_\mathrm{ngb}} m_i \frac{X_i}{\rho_i}\ W\left(|\vec{r}_0 - \vec{r}_i|, h\right)\ . \end{aligned} \tag{2.14}$$

From (2.14) we can directly calculate gradients by making use of the circumstance that only the kernel function W depends on our target point $\vec{r}_0$:

$$\begin{aligned} \nabla X(\vec{r}_0) &= \nabla \sum_i^{N_\mathrm{ngb}} m_i \frac{X_i}{\rho_i}\ W\left(|\vec{r}_0 - \vec{r}_i|, h\right) \\ &= \sum_i^{N_\mathrm{ngb}} m_i \frac{X_i}{\rho_i}\ \nabla W\left(|\vec{r}_0 - \vec{r}_i|, h\right) \end{aligned} \tag{2.15}$$

While Eqs. (2.12), (2.14), and (2.15) are valid for any target point within our simulation domain, we will almost exclusively use them to evaluate quantities at those positions where a particle is placed, effectively calculating the properties associated to this particle. This process is illustrated in Fig. 2.1.

2.2.2 Equations of Motion

At this point, we have established how to evaluate a Lagrangian fluid at any given point in time, making use of a kernel function for interpolation. What we are lacking still is a way to evolve the system in time. We will shorten the following derivation for readability and repeat our advise for the interested reader to consult Price (2012) for a thorough treatment.

The equations of motion are a direct consequence of the principle of least action, stating

$$\int \delta L \, \mathrm{d}t = 0 \tag{2.16}$$

where L denotes the discrete Lagrangian ($L = T - V$ with the kinetic energy T and the potential energy V) and δ is an arbitrary, infinitesimal variation of the particle coordinates in phase space. Making use of the generality of δ in Eq. (2.16), we find the Euler-Lagrange equation:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \vec{v}_j}\right) - \frac{\partial L}{\partial \vec{r}_j} = 0\ . \tag{2.17}$$

In our purely hydrodynamical examination, the potential energy is equal to the thermal energy of our particles and we can equate

$$L = \sum_i m_i \left(\frac{1}{2}\vec{v}_i^{\,2} - u_i\right) \tag{2.18}$$

with the specific internal energy $u_i = u_i(\rho_i, s_i)$, s_i being the specific entropy. We must now evaluate the two derivatives of L that appear in Eq. (2.17). In doing so, we make

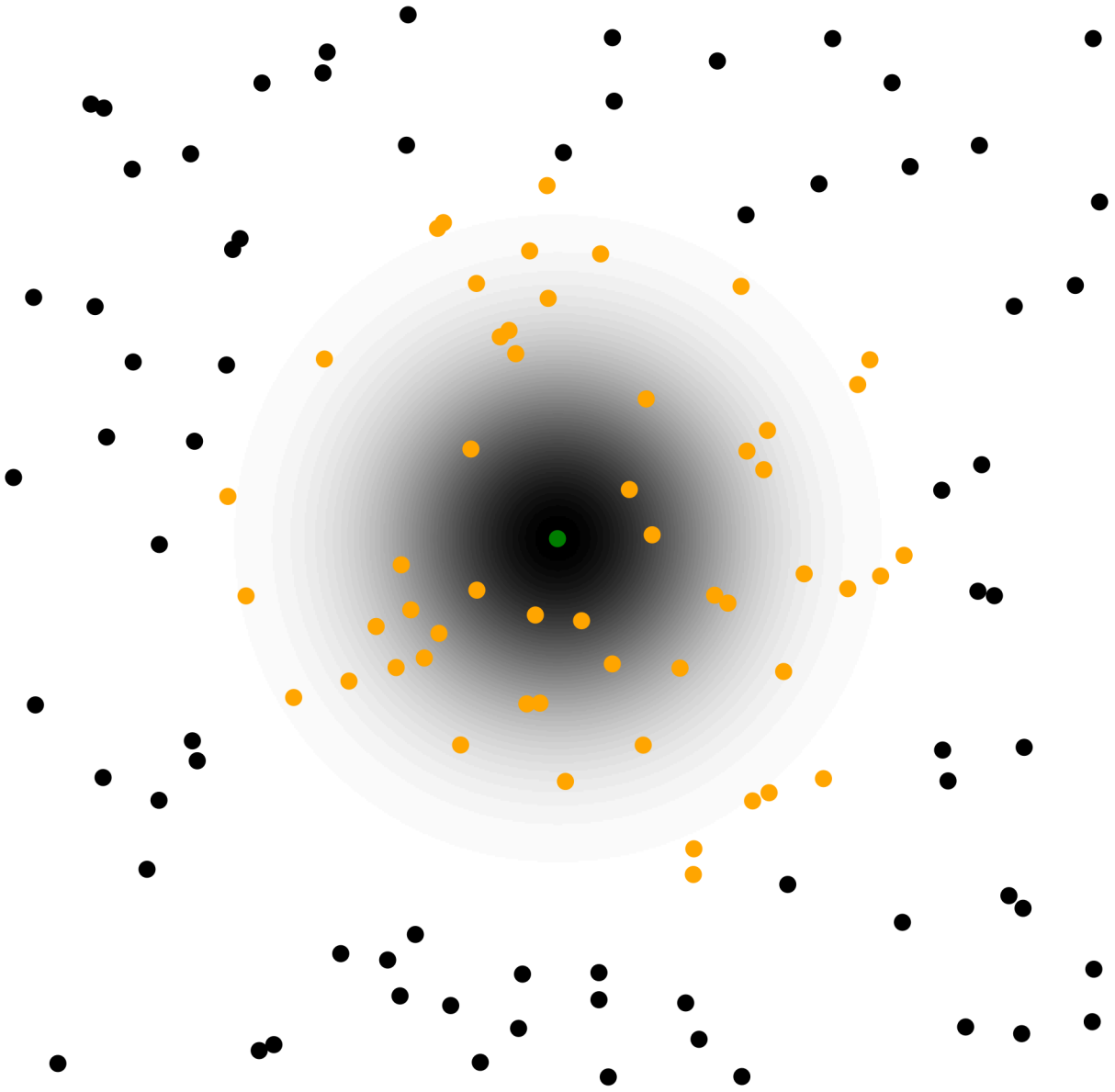


Figure 2.1: Visualization of the kernel averaging as done in SPH, using a random particle distribution. In this case, the grey gradient corresponds to the M^4 kernel, see Eq. (2.23). Due to the smooth transition of the kernel function towards 0 near $|\vec{r}_i - \vec{r}_j|/h \to 1$, the edges are not easily distinguishable. For better visibility, all particles that still lie within the kernel of the target particle (green) are coloured orange.

use of the first law of thermodynamics (Eq. 2.8) and also take into account that h is a function of ρ^2 with $\partial h/\partial\rho = -h/(\rho d)$ with dimensionality d. We then find

$$\frac{\mathrm{d}\vec{v}_j}{\mathrm{d}t} = -\sum_i m_i \left[\frac{p_j}{\Omega_j \rho_j^2} \frac{\partial W\left(\left|\vec{r}_j - \vec{r}_i\right|, h_j\right)}{\partial \vec{r}_j} + \frac{p_i}{\Omega_i \rho_i^2} \frac{\partial W\left(\left|\vec{r}_j - \vec{r}_i\right|, h_i\right)}{\partial \vec{r}_j} \right] \tag{2.19}$$

[2]This is not an inherent fact of the theory as introduced so far but rather a convenient choice, as will become clear later.

with $h_j = h(\vec{r}_j)$ and

$$\Omega_j = \left[1 - \frac{\partial h_j}{\partial \rho_j} \sum_i m_i \frac{\partial W\left(\left|\vec{r}_j - \vec{r}_i\right|, h_j\right)}{\partial h_j} \right] . \tag{2.20}$$

Note that in deriving Eq. (2.19) we implicitly assumed L to be differentiable in the first place. We thus imposed the condition that our solution cannot feature any discontinuities. In physical reality, this assumption is, of course, not always justified, e.g. if shocks are present in our system. We will come back to this problem in later sections.

One last ingredient we need for being able to advance our simulation in time is an equation that describes the temporal evolution of energy. In the absence of dissipation, the specific internal energy changes as

$$\frac{\mathrm{d}u_j}{\mathrm{d}t} = \frac{p_j}{\rho_j^2} \frac{\mathrm{d}\rho_j}{\mathrm{d}t} . \tag{2.21}$$

Combining this with an expression for the density ρ_j analogue to Eq. (2.12) yields us then

$$\frac{\mathrm{d}u_j}{\mathrm{d}t} = \frac{p_j}{\Omega_j \rho_j^2} \sum_i m_i \left(\vec{v}_j - \vec{v}_i\right) \nabla_j W\left(\left|\vec{r}_i - \vec{r}_j\right|, h_j\right) . \tag{2.22}$$

In principle, SPH schemes allow for other, equivalent quantities (e.g. specific total energy or entropy) to be traced instead of the internal energy and many codes make use of this free choice. We decided to discuss the internal energy here for no other reason than the simplicity of the derivation of Eq. (2.22) as well as the obvious correspondence to Eq. (2.10).

With Eqs. (2.12), (2.19), and (2.22) (or equivalent) we now have the basic description we need to realistically evolve a fluid in time. Furthermore, upon closer inspection, we can identify the three equations as corresponding to the continuity equation (Eq. 2.4), Euler's equation (Eq. 2.7), and the hydrodynamical energy equation (Eq. 2.10), respectively, demonstrating that the SPH approach is indeed a means to simulate hydrodynamical interactions. The next sections will highlight some peculiarities one has to take into account when applying SPH as well as briefly discuss the implementation of magnetic fields.

2.2.3 Kernel Functions

The choice of the right kernel function W for a simulation is of vital importance due to its prevalence in almost all relevant SPH calculations. All functions that might be considered should fulfil a set of prerequisites:

- The function must be monotonically decreasing. This requirement is quite intuitive since we expect close neighbours to impact a target particle's properties much more than particles further away.
- The function must be radially symmetric, i.e. $W(\vec{r} - \vec{r}') = W(|\vec{r} - \vec{r}'|)$. All other choices would lead to an unmotivated directional bias.

- The function must be normalized as $\int_V W(\vec{r}-\vec{r}',h)\,\mathrm{d}V' = 1$ (V being the total simulation domain) so as to fulfil basic conservation properties. This requirement is, of course, not as constraining since any otherwise qualified function can simply be renormalized.
- The function must have a flat slope in the central part to avoid small changes in neighbour positions resulting in large property changes.
- Lastly, the function must feature a smooth transition $W \to 0$ as $|\vec{r}-\vec{r}'| \to h$ in order to, once again, prevent small changes in neighbour position (marginally inside/outside of the kernel) leading to significant changes in W.

A popular choice for a kernel function is the cubic spline that was suggested in Monaghan (1992). We will refer to this spline as the M^4 kernel in the following. Here, for comparability with other kernel functions to be quoted later on, it is reformulated to account for a maximum extend of h (instead of $2h$ as in the original formulation):

$$W(\vec{r},h) = \frac{\mathcal{C}_d}{h^d}\begin{cases} 1+6\left(\frac{|\vec{r}|}{h}-1\right)\left(\frac{|\vec{r}|}{h}\right)^2 & \text{if } 0 \le \frac{|\vec{r}|}{h} \le 0.5 \\ 2\left(1-\frac{|\vec{r}|}{h}\right)^3 & \text{if } 0.5 < \frac{|\vec{r}|}{h} \le 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.23}$$

with d the number of dimensions and $\mathcal{C}_1 = 4/3$, $\mathcal{C}_2 = 40/(7\pi)$, and $\mathcal{C}_3 = 8/\pi$.

More recently, Dehnen and Aly (2012) were able to show that convergence at shear flows, a notorious problem for kernels such as the previously discussed M^4, can be achieved much more easily when using the Wendland functions (after Wendland 1995) as a kernel. Rosswog (2015) found similar results. The second highest order function of these is now commonly referred to as the C^6 kernel:

$$W(\vec{r},h) = \begin{cases} \frac{55}{32h}\left(1-\frac{|\vec{r}|}{h}\right)^7\left(1+7\frac{|\vec{r}|}{h}+19\left(\frac{|\vec{r}|}{h}\right)^2+21\left(\frac{|\vec{r}|}{h}\right)^3\right) & \text{if } 0 \le \frac{|\vec{r}|}{h} \le 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.24}$$

in one dimension and

$$W(\vec{r},h) = \frac{\mathcal{C}_d}{h^d}\begin{cases} \left(1-\frac{|\vec{r}|}{h}\right)^8\left(1+8\frac{|\vec{r}|}{h}+25\left(\frac{|\vec{r}|}{h}\right)^2+32\left(\frac{|\vec{r}|}{h}\right)^3\right) & \text{if } 0 \le \frac{|\vec{r}|}{h} \le 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.25}$$

in two or three dimensions ($\mathcal{C}_2 = 78/(7\pi)$, $\mathcal{C}_3 = 1365/(64\pi)$).

In addition to the choice of the kernel function, the smoothing length h should also be chosen with intent. In practice, h is determined dynamically such that a fixed number

of neighbours is achieved:

$$N_{\mathrm{Ngb}} \stackrel{!}{=} \sum_{i}^{N^*_{\mathrm{Ngb}}} W(\vec{r} - \vec{r}_i, h) \ . \tag{2.26}$$

Note that here, N^*_{Ngb} refers to the real number of particles inside the kernel, i.e. $N^*_{\mathrm{Ngb}} \in \mathbb{N}$, whereas N_{Ngb} is the sum of these particles, multiplied by their individual kernel weight, hence $N_{\mathrm{Ngb}} \in \mathbb{R}$[3,4]. This method of determining h has the added benefit of making the resolution of SPH automatically adaptive — in dense regions, a small smoothing length is chosen, making the interpolations intrinsically more accurate. In sparsely populated, uneventful regimes on the other hand, a rough estimate with a larger value of h suffices in most cases. In the remainder of this work, the term *neighbour number* shall always refer to the kernel weighted sum on the left side of Eq. (2.26).

2.2.4 Magnetohydrodynamics

Magnetic fields have been known to have significant impacts on astrophysical systems of various scales for quite some time (see e.g. Parker 1958, Turner and Widrow 1988, and Brandenburg and Subramanian 2005; also refer to Widrow 2002 for an extensive, if slightly old review). Luckily, SPH schemes can be adjusted in a straightforward manner such that they include the treatment of magnetohydrodynamics (MHD) within the fluid.

From the induction equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\vec{B}}{\rho}\right) = \left(\frac{\vec{B}}{\rho} \cdot \nabla\right) \vec{v} \tag{2.27}$$

with the magnetic field vector $\vec{B}$ we can directly infer

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\vec{B}_j}{\rho_j}\right) = \sum_i m_i \left(\vec{v}_i - \vec{v}_j\right) \frac{\vec{B}_j}{\Omega_j \rho_j^2} \nabla W\left(\vec{r}_j - \vec{r}_i, h_j\right) \ . \tag{2.28}$$

The equations of motion can be derived via a similar ansatz as in Sec. 2.2.2 with a slightly modified Lagrangian:

$$L = \sum_i m_i \left(\frac{1}{2}\vec{v}_i^2 - u_i - \frac{1}{2\mu_0}\frac{B_i^2}{\rho_i}\right) \tag{2.29}$$

with the vacuum permeability μ_0. We will skip some intermediate steps for brevity since they do not yield much insight. The final result for the acceleration equation reads

$$\frac{\mathrm{d}v_j^k}{\mathrm{d}t} = \sum_i m_i \left[\frac{S_j^{kl}}{\Omega_j \rho_j^2} \nabla_j^l W\left(\vec{r}_j - \vec{r}_i, h_j\right) + \frac{S_i^{kl}}{\Omega_i \rho_i^2} \nabla_j^l W\left(\vec{r}_j - \vec{r}_i, h_i\right)\right] \tag{2.30}$$

where k and l run over the spatial dimensions and S^{kl} is the MHD stress tensor:

$$S^{kl} = -\left(p + \frac{1}{2\mu_0} B^2\right) \delta^{kl} + \frac{1}{\mu_0} B^k B^l \ . \tag{2.31}$$

[3]Compare this to Fig. 2.1 where 53 neighbours have been associated with the target particle, even though the kernel weighted neighbour number only equals to roughly ten, i.e. $N^*_{\mathrm{Ngb}} = 53$, $N_{\mathrm{Ngb}} \approx 10$.

[4]In practice, the requirement of Eq. (2.26) to be an exact equality is slightly lifted by setting an allowed deviation. This allows the root-finder that calculates the required h to terminate much faster.

The energy equation is altered in a similar manner:

$$\frac{\mathrm{d}u_j}{\mathrm{d}t} = \sum_i m_i \left(\frac{S_j^{kl}}{\Omega_j \rho_j^2} v_i^k \nabla_j^l W\left(\vec{r}_j - \vec{r}_i, h_j\right) + \frac{S_i^{kl}}{\Omega_i \rho_i^2} v_j^k \nabla_j^l W\left(\vec{r}_j - \vec{r}_i, h_i\right) \right) \tag{2.32}$$

The equation governing the density (Eq. 2.12) is not altered by the addition of magnetic fields.

One problem in all numerical treatments of magnetic fields is to ensure that the absence of magnetic monopoles ($\nabla \cdot B = 0$) is respected in all cases. This behaviour is not inherent in the equations derived above and therefore, fixing mechanisms must be invoked. Popular choices are to use a more accurate but non-conservative gradient in the anisotropic force, or to subtract either the unphysical source term or a constant from the stress (see also Sec. 4.5).

2.2.5 Correction Terms

While SPH offers a convenient way to formulate and solve hydrodynamical problems numerically, it has some known flaws. A part of which is inherent and our motivation to implement a new scheme (see Sec. 3.1) but some can be mitigated by expanding the framework as discussed this far. We will present a small, non-exhaustive subset of these corrective terms here.

∇h terms Hernquist (1993) demonstrated that while standard SPH approaches are quite good at conserving energy, entropy conservation is everything but guaranteed. As mentioned before, we are free to exchange the internal energy for the entropy as the energetic variable we want to evolve. This does indeed lead to significantly better entropy conservation but instead, the conservation of energy is lost in turn. To overcome this problem, Springel and Hernquist (2002) introduced a tweak to the equation of motion that fundamentally improves the synchronous conservation of the two quantities:

$$\frac{\mathrm{d}\vec{v}_j}{\mathrm{d}t} = -\sum_i m_i \left[f_j \frac{p_j}{\Omega_j \rho_j^2} \frac{\partial W\left(\left|\vec{r}_j - \vec{r}_i\right|, h_j\right)}{\partial \vec{r}_j} + f_i \frac{p_i}{\Omega_i \rho_i^2} \frac{\partial W\left(\left|\vec{r}_j - \vec{r}_i\right|, h_i\right)}{\partial \vec{r}_j} \right] \tag{2.33}$$

with the corrective terms

$$f_j = \left(1 + \frac{h_j}{3\rho_j} \frac{\partial \rho_j}{\partial h_j} \right)^{-1} . \tag{2.34}$$

Note that this correction implicitly assumes that a constant mass is present inside the kernel with radius h (instead of a constant neighbour number as we stated in Eq. 2.26). However, if we assume constant and equal particle masses, these two constraints are, of course, the same.

Artificial viscosity In highly dynamical regions such as shocks, dissipative terms must be taken into account to correctly reproduce the dispensation of energy. But, since SPH is constructed in a way to solve the ideal hydrodynamical equations, these effects are not considered by default. One (of many) ways to implement an artificial viscosity term is given in Monaghan (1992) where the new equation of motion reads:

$$\frac{\mathrm{d}\vec{v}_j}{\mathrm{d}t} = -\sum_i m_i \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} + \Pi_{ij} \right) \nabla_j W\left(\vec{r}_j - \vec{r}_i, h_j\right) \tag{2.35}$$

with

$$\Pi_{ij} = \begin{cases} \dfrac{-\alpha \overline{c_{ij}} \mu_{ij} + \beta \mu_{ij}^2}{\overline{\rho_{ij}}} & \text{if } \left(\vec{v}_j - \vec{v}_i\right) \cdot \left(\vec{r}_j - \vec{r}_i\right) < 0 \\ \\ 0 & \text{otherwise} \end{cases} \tag{2.36}$$

and

$$\mu_{ij} = \frac{h_{ij} \left(\vec{v}_j - \vec{v}_i\right) \left(\vec{r}_j - \vec{r}_i\right)}{\left(\vec{r}_j - \vec{r}_i\right)^2 + \eta_{ij}^2} \ . \tag{2.37}$$

The condition $(\vec{v}_j - \vec{v}_i) \cdot (\vec{r}_j - \vec{r}_i) > 0$ is equal to $\nabla \vec{v} > 0$ and thus the viscous term vanishes in this case. The parameters α and β can be adjusted depending on the individual needs of the simulation at hand, but the authors report good results for $\alpha \approx 1$ and $\beta \approx 2$. η_{ij} prevents singularities in high density regions and is best chosen as $\eta_{ij} \approx h_{ij}/10$. Eq. (2.35) conserves total linear and angular momenta and yields both shear and bulk viscosities (linear term) as well as Von Neumann-Richtmeyer viscosity (quadratic term).

Artificial conduction In order to improve mixing properties of SPH implementations, various authors have proposed schemes to facilitate transport of internal energy (see e.g. Price 2008, Wadsley et al. 2008, or Read and Hayfield 2012). One of the more recent efforts can be found in Beck et al. (2016) who approximate the conduction as

$$\left.\frac{\mathrm{d}u_j}{\mathrm{d}t}\right|_{\mathrm{cond}} = \sum_i \frac{2m_i}{\rho_j + \rho_i} \left(u_i - u_j\right) \alpha_{ij}^c \overline{F_{ij}} \sqrt{\frac{2\left|p_j - p_i\right|}{\rho_j + \rho_i}} \tag{2.38}$$

with $\overline{F_{ij}} = (F_{ij}(h_i) + F_{ij}(h_j))/2$ the symmetrized scalar part of the kernel gradient terms $\nabla_i W(\vec{r}_i - \vec{r}_j, h_i) = F_{ij} \vec{r}_{ij} / |\vec{r}_{ij}|$ and the symmetrized conduction coefficient

$$\alpha_{ij}^c = \frac{1}{2} \left(\frac{h_i}{3} \frac{|\nabla u|_i}{|u_i|} + \frac{h_j}{3} \frac{|\nabla u|_j}{|u_j|} \right) \ . \tag{2.39}$$

2.3 Eulerian Methods

Eulerian methods follow a vastly different paradigm than their Lagrangian counterparts. Whereas we previously represented the fluid of consideration as an ensemble of particles with known masses and positions, we now instead simulate a grid (either static or dynamic) occupied by the fluid. Hence, Eulerian schemes are often referred to as grid codes or grid schemes and we will use these terms synonymously. Inside each cell, the properties of the fluid are (at a given point in time) described independently. The equations of motion are replaced by fluxes from one adjacent cell to another.

While the field of Eulerian methods is equally wide and complex as that of Lagrangian methods, we will restrict ourselves to a very short overview of which, closely following Teyssier (2015), as the focus of this work is on the comparison of two Lagrangian methods.

2.3.1 Simulating Fluxes

The starting point for grid schemes is the reformulation of the equations derived in Sec. 2.1 in their Eulerian form, highlighting conservation properties more prominently:

$$\frac{\partial \rho \vec{v}}{\partial t} + \nabla \left(\rho \vec{v} \times \vec{v} + p \mathbb{1} \right) = 0 \tag{2.40}$$

$$\frac{\partial \rho}{\partial t} + \nabla \left(\rho \vec{v} \right) = 0 \tag{2.41}$$

$$\frac{\partial E}{\partial t} + \nabla \left(\vec{v} \left(E + p \right) \right) = 0 \tag{2.42}$$

where we replaced the internal energy u with the fluid energy density $E = \rho(u + \frac{1}{2}v^2)$. The advantage of the above formulations is that we can forego a flaw of the Lagrangian equivalents that we thus far only touched upon very briefly — they are only viable in continuous environments. While Lagrangian schemes have to apply corrective functions to reduce the resulting inaccuracies when dealing with discontinuities, some of which we mentioned in Sec. 2.2.5, Eulerian schemes elegantly avoid these problems altogether. In practice, this means that the flow variables (that are exchanged between cells) can be discontinuous from one cell to another, as long as the flux functions (that appear in the divergence operator in Eqs. 2.40 - 2.42) remain continuous.

To solve the above equations in the presence of propagating waves with sound speed

$$c_\mathrm{s} = \sqrt{\frac{\gamma\, p}{\rho}} \ , \tag{2.43}$$

a popular numerical technique is the one brought forward by Godunov and Bohachevsky (1959). In the following, we will express the fluid quantities in terms of vectors of conservative variables

$$\vec{U} = \vec{U}\left(\vec{r}, t\right) := \begin{pmatrix} \rho \\ \rho \vec{v} \\ E \end{pmatrix} \ . \tag{2.44}$$

Each cell is assigned a cell-averaged conservative vector

$$\vec{U}_i^n = \frac{1}{V_i} \int_{V_i} \vec{U}\left(\vec{r}, t^n\right) \ \mathrm{d}V \tag{2.45}$$

with n indicating the discrete time step and V_i being the volume of the ith cell. The time evolution of each cell can then be described as

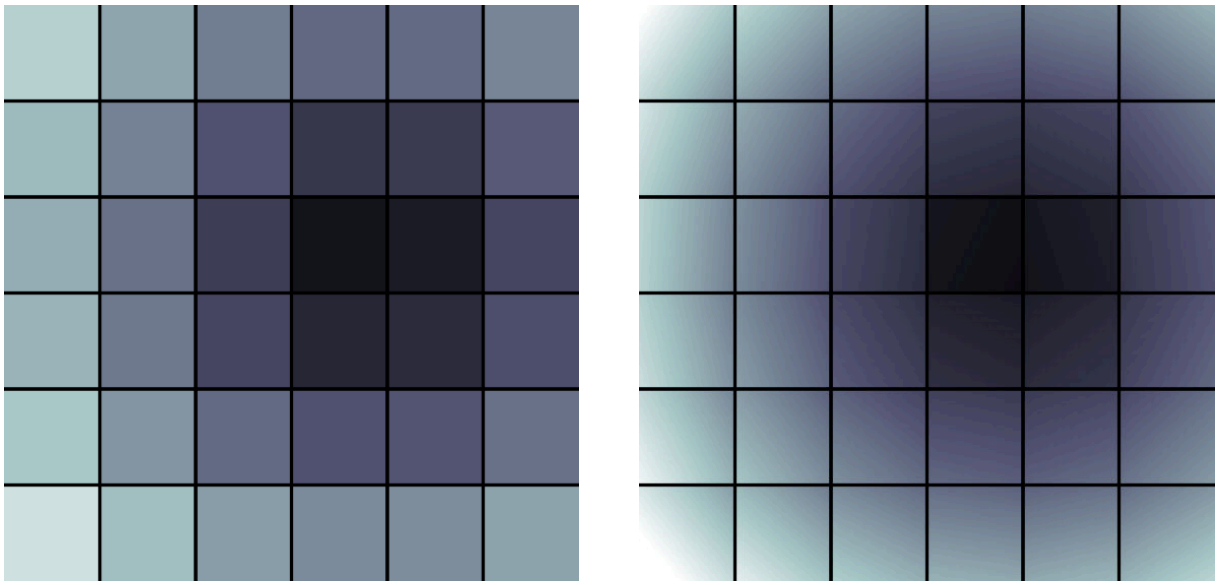
$$\frac{\vec{U}_i^{n+1} - \vec{U}_i^n}{\Delta t} = -\frac{1}{V_i} \int_{S_i} \left\langle \vec{F} \right\rangle_t \cdot \vec{n} \ \mathrm{d}S \tag{2.46}$$

where $\vec{n}$ is the normal vector of the cell face S_i and $\left\langle \vec{F} \right\rangle_t$ is the time-averaged flux function

$$\left\langle \vec{F} \right\rangle_t = \frac{1}{\Delta t} \int_{t^n}^{t^{n+1}} \vec{F}\left(\vec{r}, t\right) \ \mathrm{d}t \ . \tag{2.47}$$

We stress that the above formulations are agnostic to the exact implementations of both $\vec{U}(\vec{r}, t)$ and $\vec{F}(\vec{r}, t)$. The former can be used to change the physical representation

within a cell — in so-called first-order Godunov schemes, $\vec{U}$ is assumed to be constant at each point within a cell, whereas higher order approaches assume linear or parabolic gradients, see Fig, 2.2 for a comparison. The latter is used to fine-tune the accuracy of quantities exchanged between cells. In practice, Riemann solvers are applied at each cell boundary to calculate the flux functions and whether exact or approximate variants are used is left to be decided by the simulator. We will deepen our discussion of Riemann solvers in Sec. 3.3.



(a) First order **(b)** Second order

Figure 2.2: Comparison between Godunov schemes of first order (a) and second order (b). The colour indicates the value of an arbitrary quantity at each point within the simulation domain. The second order scheme produces a constant gradient over each cell. Note that in the middle (around the darkest cells) the transition between cells is not approximated smoothly due to the implementation of slope limiters. We shall discuss their relevance in more detail in Sec. 3.2.

The theory laid out so far represents the bare backbone of grid codes and any competitive implementation must take into account plenty of additional effects such as non-ideal hydrodynamics (see e.g. Fromang et al. 2007), gravity (e.g. Truelove et al. 1997 and Käppeli and Mishra 2014), and radiative transport (see Mihalas and Mihalas 1984 for a theoretical treatment). We refrain from delving into these in more detail here, refer the interested reader to the literature cited in this section, and shall close our brief discussion by taking a closer look at two optional features of grid codes concerning grid geometry.

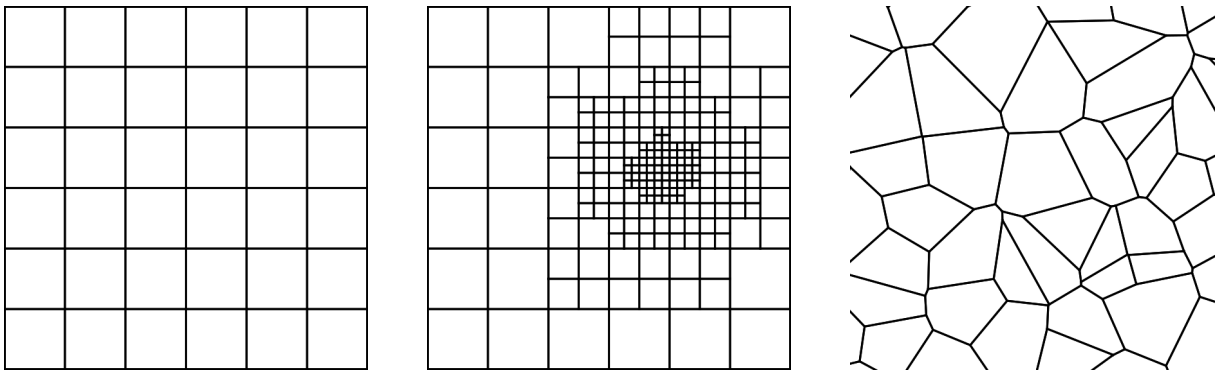
2.3.2 Adaptive Mesh Refinement and Moving Mesh

While a regular, Cartesian grid is a logical first choice for a Eulerian scheme, there is no need to restrict ourselves to such a realization. In fact, choosing a grid with stiff axes is prone to produce preferred directions along these since all cell faces will lie in perpendicular planes. Luckily, the formalism presented in Sec. 2.3.1 is valid for (almost) arbitrary grid geometries. We can utilize this circumstance in two different ways.

Adaptive Mesh Refinement While adaptive mesh refinement (AMR) codes still adhere to a Cartesian grid with fixed axes, they make use of the fact that there is no inherent need for cells to be of equal size. Similar to how SPH (automatically) adapts is resolution in very dense regions by having a smaller kernel width and thus more local smoothing, AMR schemes decrease the cell size if the density exceeds a certain threshold[5]. In 1D, this is usually done by simply halving the cell in the middle, in 2D every quadrant becomes a new cell, and in 3D every octant. This process can be repeated indefinitely until the required resolution is reached. In addition to increasing the spatial resolution, most AMR codes also increase the temporal resolution of refined cells by halving the time step Δt with each level of refinement. Brought forward by Berger and Oliger (1984) and Berger and Colella (1989), this technique is now employed in many popular astrophysical grid codes such as `RAMSES` (Teyssier, 2002) and `ENZO` (Bryan et al., 2014).

Moving Mesh To minimize numerical diffusion, so-called moving mesh or adaptive mesh redistribution schemes change the structure of the underlying grid with time. The total number of cells generally stays constant and the geometry is always chosen in such a way as to reduce the maximum wave speed. It has been shown, however, that failing to limit the resulting mesh geometry at all can lead to large numerical errors in areas of the computation domain where cell sizes vary largely (see e.g. Katz and Sankaran 2011). A contemporary code making use of the moving mesh technique is `AREPO` (Weinberger et al., 2019).

A comparison between a simple Cartesian grid, an AMR realization and a moving mesh is depicted in Fig. 2.3.



(a) Simple Cartesian grid **(b)** Cartesian with AMR **(c)** Moving mesh

Figure 2.3: Comparison of different grid geometries. The AMR pattern in panel (b) resembles the refinement around an over-dense region, e.g. a globular cluster or a galaxy. Here, a maximum refinement of three levels was chosen. The moving mesh in panel (c) is realized using a Voronoi tessellation, similar to `AREPO`'s approach.

[5]In theory, the criterion that determines whether or not a cell shall be refined can be chosen arbitrarily, the density (i.e. total mass in a cell) just being one possibility. Another popular choice is using Richardson-type estimates of the truncation error (Berger and Oliger, 1984).

3 Meshless Methods

Over the last decades, the methods for solving CFD that we introduced in the previous section have been both studied and applied extensively, yielding a plethora of scientific insight. More recently, however, a new type of solvers was developed in the field of computer science by Lanson and Vila (2008a,b) and first applied in an astrophysical context by Gaburov and Nitadori (2011). In this section, we shall highlight the key aspects of these so-called meshless methods. Since `GADGET`, the code we are working with, is currently using an SPH scheme, we will start by motivating the implementation of the new solver by drawing attention to the problems that SPH solvers are often facing[6].

3.1 The Shortcomings of SPH

3.1.1 Dealing with Discontinuities

A consequence of the kernel-based averaging which is at the very centre of the SPH approach is that large differences in particle properties are smoothed over. In most cases, this helps to reduce numerical noise since the effect that outliers have on their environments is dampened. When dealing with discontinuities, however, this smoothing fails to accurately reproduce the sharp difference between the two states. More precisely, the discontinuity is smeared out over $2h$ — one full kernel width. Fig. 3.1 illustrates this effect. The issue becomes especially problematic when other quantities such as the internal energy are injected externally (as is e.g. the case when simulating supernovae in SPH) and still feature a real (i.e. sharp) numerical discontinuity, leading to an inconsistent treatment and resulting numerical artefacts, see e.g. Price (2008).

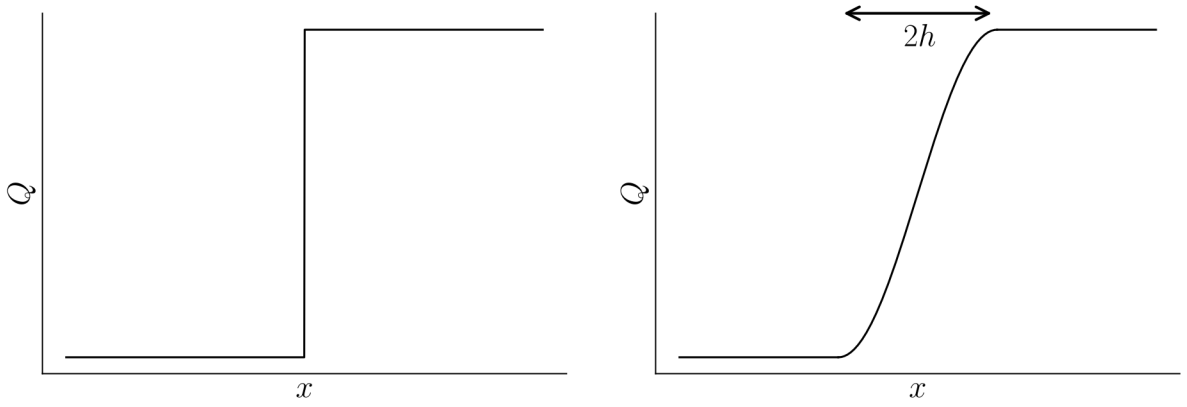


Figure 3.1: Comparison of a discontinuity of an arbitrary quantity Q in its analytic form (left panel) and when simulated using SPH (right panel).

3.1.2 Fluid Mixing

Agertz et al. (2007) carried out a study directly comparing Eulerian to Lagrangian schemes. To avoid bias caused by specific implementations, they took five AMR and

[6]It should be noted that many authors have brought forward suggestions how these problems might be remedied through various tweaks of the SPH implementation, see e.g. Hopkins (2013) and references therein. While most of those ideas were actually able to circumvent the issues they set out to fix, they always introduced other spurious effects such as additional ad hoc dissipation terms, violations of conservation laws, or reduced numerical stability.

two SPH codes into account. Despite the large number of simulations, their results were quite conclusive: Both SPH schemes failed to realistically resolve fluid mixing in a range of standard tests that were performed. Whenever two different gas phases interacted, they would remain separate even after multiple dynamical times. The grid codes fared considerably better. The authors attributed this feature to erroneous pressure forces being introduced in SPH codes inside kernels that comprise a steep density gradient. As a result, crossing the gap between high-density and low-density is impeded and most particles remain in their respective phase, hindering fluid mixing altogether. Okamoto et al. (2003) found similar results.

3.1.3 Runtime

Of course, we would be ill-advised to claim that SPH codes are running slowly per se. However, following the results of Dehnen and Aly (2012), we know that in order to reduce noise at shear flows, SPH schemes ought to use one of the Wendland kernels. Not only are they expensive due to their high order polynomials[7], they also demand a large number of neighbours. The C^6 kernel (Eq. 2.25), for example, requires ~ 300 neighbours in 3D to produce good results. In addition to the sums becoming larger and taking longer to evaluate, this also produces considerable overhead in parallel computing since, very often, not all of these particles will be evaluated on the same CPU.

3.2 Fluxes in a Lagrangian Scheme

To address the issues discussed above while still maintaining the undisputed advantages of SPH-like schemes (such as exact conservation properties, numerical robustness, automatically adaptive resolution, or the straightforward implementation in N-body codes), meshless methods maintain an overall Lagrangian approach while borrowing convenient features from grid codes.

The following derivation is a condensed version of the arguments brought forward by Hopkins (2015) and we recommend the interested reader study this publication for more details, including a comprehensive appendix.

The basis for this novel approach is the same we used in Sec. 2.3.1, see Eqs. (2.40) - (2.42). We will summarize these equations describing the nature of hydrodynamics in their Eulerian form in one single formula, making use of vector notation:

$$\frac{\partial \vec{U}}{\partial t} + \nabla \left(\vec{F} - \vec{v}_{\text{frame}} \times \vec{U} \right) = 0 \tag{3.1}$$

with $\vec{U}$ the conserved state vector as defined in Eq. (2.44), $\vec{v}_{\text{frame}}$ the velocity of an arbitrary moving frame and the flux vector

$$\vec{F} = \begin{pmatrix} \rho \vec{v} \\ \rho \vec{v} \times \vec{v} + p \mathbb{1} \\ (E + p)\, \vec{v} \end{pmatrix} . \tag{3.2}$$

[7]While one evaluation is, of course, not very time consuming, note that the kernel function is called excessively during the simulation, being present in every major equation in Sec. 2.2.

3.2.1 Domain Partitioning

We shall solve Eq. (3.1) with the Galerkin method which can be treated as a generalization of Godunov's method (Ladonkina and Tishkin, 2015). In doing so, we multiply Eq. (3.1) by an arbitrary test function $\phi = \phi(\vec{r}, t)$, integrate over the domain Ω and, after performing an integration by parts and making the (very reasonable) assumption that fluxes vanish at infinity, ultimately find:

$$0 = \frac{\mathrm{d}}{\mathrm{d}t} \int_\Omega \vec{U}\left(\vec{r}, t\right) \phi \, \mathrm{d}^d \vec{r} = \int_\Omega \vec{F}\left(\vec{U}, \vec{r}, t\right) \cdot \nabla\phi \, \mathrm{d}^d \vec{r} \tag{3.3}$$

with the co-moving derivative $\mathrm{d}f/\,\mathrm{d}t \equiv \partial f/\partial t + \vec{v}_{\mathrm{frame}}(\vec{r}, t) \cdot \nabla f$ and d the number of spatial dimensions. The freedom we now have is to decide how to associate each particle i at coordinates $\vec{r}_i$ with a discrete domain volume Ω_i. Choosing to associate every particle with its respective Voronoi cell[8] would yield a moving mesh scheme (see Sec. 2.3.2). Instead, we now choose an association that is more akin to an SPH approach: Each point $\vec{r}^*$ within the computation domain is partitioned among surrounding particles with a weighting function, prioritizing close particles. The fraction ψ_i that each point contributes to particle i can then be estimated as

$$\psi_i\left(\vec{r}^*\right) = \frac{W\left(\left|\vec{r}^* - \vec{r}_i\right|, h\right)}{\sum_j W\left(\left|\vec{r}^* - \vec{r}_j\right|, h\right)} \tag{3.4}$$

where the denominator ensures that $\sum_i \psi_i \equiv 1$. The weighting function W imposes all requirements we listed in Sec. 2.2.3 and as a result, the kernel functions we discussed therein are an excellent fit for our needs[9]. Note that the edge case of a moving mesh we discussed above corresponds to the steep limit where W converges to a δ-function

We can now insert this volume partitioning approach into Eq. (3.3) and after a few steps we find

$$0 = \sum_i \left[\phi_i \frac{\mathrm{d}}{\mathrm{d}t}\left(V_i \vec{U}_i\right) - V_i \vec{F}_i \left(\nabla\phi\right)_{\vec{r}=\vec{r}_i}\right] \tag{3.5}$$

with the effective volume

$$V_i = \int \psi_i\left(\vec{r}\right) \mathrm{d}^d \vec{r} \tag{3.6}$$

representing the sum of all domain partitions that particle i received. In theory, the integral extends to spatial infinity but in practice, integrating a sphere of radius h suffices.

3.2.2 Gradient Estimates

The accuracy of meshless schemes now hinges on how well we can estimate the gradients $\nabla\phi$. A popular choice are locally-centred least-square matrix gradient operators (see e.g. Luo et al. 2008). While these can, in theory, be designed to be accurate to arbitrary orders, we will restrict ourselves to the second-order approach to maintain reasonable

[8] A particle's Voronoi cell is defined such that it contains all points within the computation domain that lie closer to it than to any other particle.

[9] Due to the inherent normalization of Eq. (3.4), we could theoretically lift the requirement for W itself to be normalized. We will, however, hold on to this attribute for better comparability to SPH.

runtime. Gradients are then estimated as

$$\begin{aligned}(\nabla f)_i^\alpha &= \sum_j \sum_{\beta=1}^{d} \left(f_j - f_i\right) \underline{\underline{B_i}}^{\alpha\beta} \left(\vec{r}_j - \vec{r}_i\right)^\beta \psi_j\left(\vec{r}_i\right) + \mathcal{O}\left(h_i^2\right) \\ &\equiv \sum_j \left(f_j - f_i\right) \chi_j^\alpha\left(\vec{r}_i\right) \end{aligned} \tag{3.7}$$

where we substituted

$$\chi_j^\alpha\left(\vec{r}_i\right) = \sum_{\beta=1}^{d} \underline{\underline{B_i}}^{\alpha\beta} \left(\vec{r}_j - \vec{r}_i\right)^\beta \psi_j\left(\vec{r}_i\right) = \underline{\underline{B_i}}^{\alpha\beta} \left(\vec{r}_j - \vec{r}_i\right)^\beta \psi_j\left(\vec{r}_i\right) \tag{3.8}$$

and used Einstein's summation convention over the element indices α and β. The matrix $\underline{\underline{B_i}}^{\alpha\beta}$ is defined in terms of another matrix $\underline{\underline{E_i}}^{\alpha\beta}$ as $\underline{\underline{B_i}} = \underline{\underline{E_i}}^{-1}$ with

$$\underline{\underline{E_i}}^{\alpha\beta} = \sum_j \left(\vec{r}_j - \vec{r}_i\right)^\alpha \left(\vec{r}_j - \vec{r}_i\right)^\beta \psi_j\left(\vec{r}_i\right) \ . \tag{3.9}$$

Inserting Eq. (3.7) in Eq. (3.5) and rearranging the sums, we find

$$0 = \sum_i \phi_i \left\{ \frac{\mathrm{d}}{\mathrm{d}t}\left(V_i \vec{U}_i\right) + \sum_j \left[V_i \vec{F}_i^\alpha \chi_j^\alpha\left(\vec{r}_i\right) - V_j \vec{F}_j^\alpha \chi_i^\alpha\left(\vec{r}_j\right)\right] \right\} \tag{3.10}$$

which must hold for any function ϕ_i, thus:

$$0 = \frac{\mathrm{d}}{\mathrm{d}t}\left(V_i \vec{U}_i\right) + \sum_j \left[V_i \vec{F}_i^\alpha \chi_j^\alpha\left(\vec{r}_i\right) - V_j \vec{F}_j^\alpha \chi_i^\alpha\left(\vec{r}_j\right)\right] \ . \tag{3.11}$$

To automatically include *real* dissipative terms, one last step we need to perform is shift the fluxes $\vec{F}_i$ from the position of the particles to inter-particle locations. Now, instead of describing the total flux of one particle, our new flux vectors F_{ij} denote the fluxes between two particles i and j alone. Conservation demands $\vec{F}_{ij} \equiv -\vec{F}_{ji}$. Using the vector $A_{ij}^\alpha = V_i \chi_j^\alpha(\vec{r}_i) - V_j \chi_i^\alpha(\vec{r}_j)$ and the shorthand $\vec{A}_{ij} = |A|_{ij} \hat{A}_{ij}$ with the projection operator $\hat{}$ we ultimately find the flux equation in the form

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(V_i \vec{U}_i\right) + \sum_j \vec{F}_{ij} \cdot \vec{A}_{ij} = 0 \ . \tag{3.12}$$

Closer inspection tells us that this expression is nothing else than a continuity equation, stating that the temporal change of conserved quantities $\vec{U}_i$ multiplied by the effective volume V_i associated with particle i must equal the sum of all fluxes to neighbouring particles j. Hence, the term $\vec{A}_{ij}$ must correspond to an effective face area between the particles i and j.

3.2.3 Solving the Flux Equation

To be able to solve Eq. (3.12), we need to perform some additional steps.

First, we must determine where we actually want to calculate the flux. In principle, any point on a straight line between $\vec{r}_i$ and $\vec{r}_j$ would be admissible. To preserve symmetry,

a natural choice is the quadrature point, i.e. the point that is partitioned equally between i and j along this line:

$$\vec{r}_{ij}^{\,\mathrm{quad}} = \vec{r}_i + \frac{h_i}{h_i + h_j}\left(\vec{r}_j - \vec{r}_i\right) \ . \tag{3.13}$$

According to Hopkins (2015), however, a much simpler approximation yields results of almost equal quality:

$$\vec{r}_{ij} = \frac{\vec{r}_i + \vec{r}_j}{2} \ . \tag{3.14}$$

This approximation is justified by the fact that neighbouring particles should have similar smoothing lengths as they experience similar densities (and since Eq. 3.14 is the limit of Eq. 3.13 for $h_i - h_j \to 0$). Now, we can also define the frame velocity we already introduced in Eq. (3.1) as

$$\vec{v}_{\mathrm{frame}} = \vec{v}_i + \left(\vec{v}_j - \vec{v}_i\right)\left[\frac{\left(\vec{r}_{ij} - \vec{r}_i\right)\cdot\left(\vec{r}_j - \vec{r}_i\right)}{\left|\vec{r}_j - \vec{r}_i\right|^2}\right] \ . \tag{3.15}$$

Fig. 3.2 illustrates the geometric reconstruction of cell faces as well as the domain partition that is performed in meshless schemes.

To maintain a framework to which we can easily attach fluxes we will trivially transform our conservative state vectors $\vec{U}$ into primitive vectors $\vec{W}$ with $\vec{W} = (\rho, \vec{v}, p)$.

Next, we must extrapolate the physical quantities defined at the locations of the particles i and j to the cell face. We already performed this exercise in Eq. (3.7) and can use the same approach here. We ought to, however, make sure that we do not overshoot with our reconstruction and introduce artificial extrema. To this avail, we must also introduce a slope limiting procedure in which we substitute $\nabla f_i \mapsto \alpha_i \nabla f_i$ with $\alpha_i \in (0, 1]$. There exist plenty of possibilities to calculate the numerical value of α_i and we refer the interested reader to appendix B in Hopkins (2015) for a general formulation of the slope limiting procedure as well as further references.

Having done so, we have obtained information of the primitive variables at the location of the face, extrapolated from both sides (i.e. both from particle i and j). We are free to rotate (and boost, as will become important in Sec. 3.4) our reference frame and associate these two vectors with a left and right state, $\vec{W}_{\mathrm{L}}$ and $\vec{W}_{\mathrm{R}}$, respectively. For a more accurate flux estimate, we can also shift these primitive vectors to their time-centred state, i.e. $\vec{W}_{\mathrm{L,R}} \mapsto (\partial \vec{W}_{\mathrm{L,R}}/\partial t)(\Delta t/2)$. Luckily, this is rather easily done with

$$\frac{\partial \vec{W}}{\partial t} = -\begin{pmatrix} \vec{v} & \rho & 0 \\ 0 & \vec{v} & 1/\rho \\ 0 & \gamma p & \vec{v} \end{pmatrix} \nabla \vec{W} \ . \tag{3.16}$$

At this point, all that is left for us to do is calculate the flux between the (now time-centred) $\vec{W}_{\mathrm{L}}$ and $\vec{W}_{\mathrm{R}}$, multiply these with the time step Δt, rotate back to the simulation frame, and update all quantities accordingly.

3.3 Riemann Solvers

As the two primitive vectors $\vec{W}_{\mathrm{L}}$ and $\vec{W}_{\mathrm{R}}$ are extrapolated independently from each other (one from particle i and its respective gradients, the other from j), we do not expect

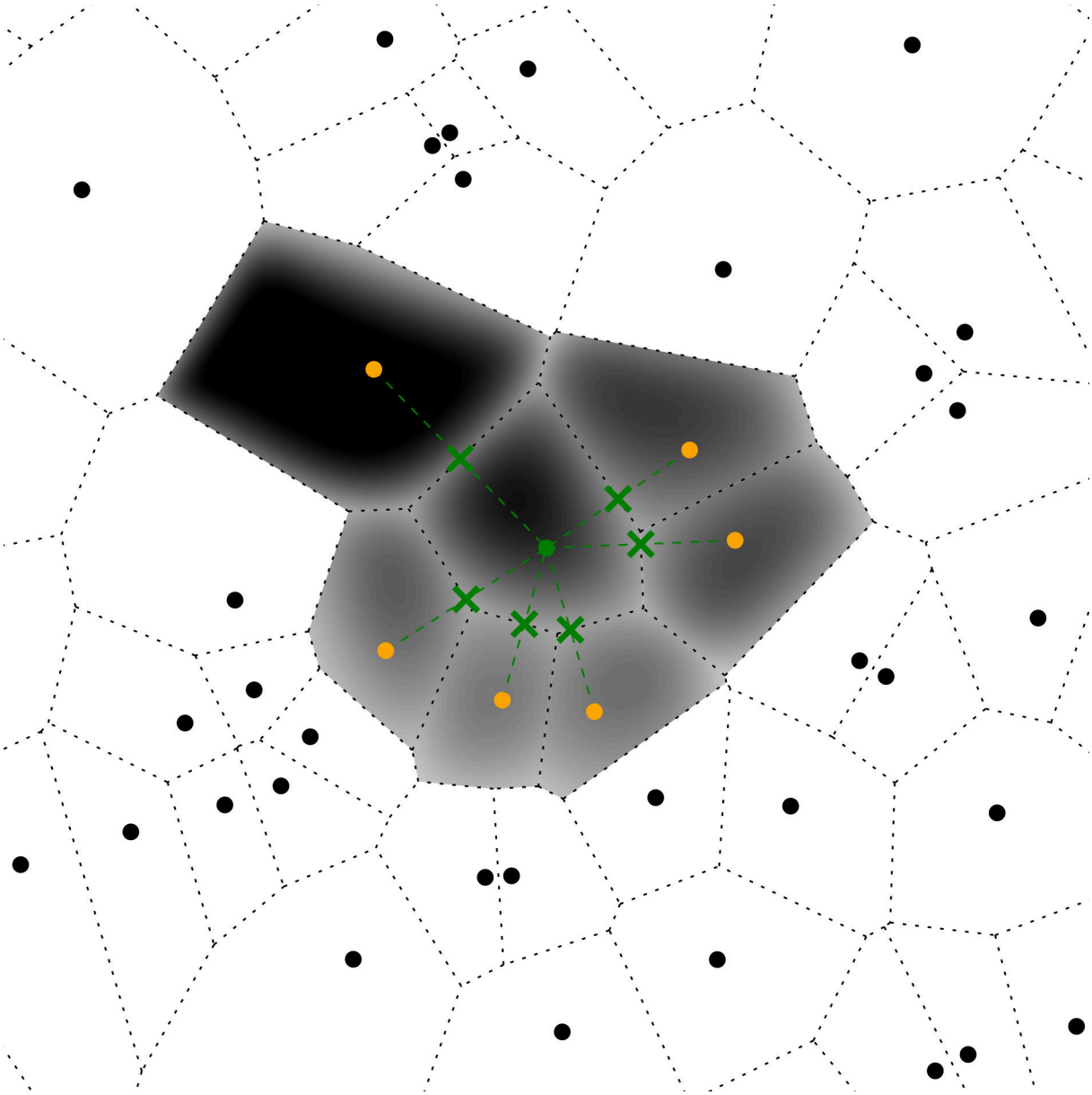


Figure 3.2: Visualization of the geometry of meshless schemes. Note that this depicts the same random particle distribution as Fig. 2.1, only zoomed in on the target particle (green dot). The orange dots represent the neighbours to the target particle with which the flux computations are performed. The grey gradients represent the domain partition around each particle. While in theory, they would extend further out (to a maximum radial distance of h), the dotted mesh confines the area in which the respective particle receives the main contribution, thereby creating a Voronoi tessellation. Also note that the faces for the flux computation (green crosses) are taken as the middle point between the respective particle pair (Eq. 3.14), thereby exactly coinciding with the edges of the Voronoi mesh. Contrary to moving mesh schemes (see Fig. 2.3c), however, it is not in fact necessary to explicitly calculate this tessellation.

---

them to be equal even though both describe fluid quantities at the exact same point. We thus need an algorithm capable of calculating fluxes (i.e. solving the continuous hydrodynamical equations) in the presence of discontinuities. Such algorithms are called Riemann solvers. Many of these exist at the time of this writing, both of exact and

approximate nature. We will highlight one example each, focussing on those that were in fact implemented in `GADGET`.

3.3.1 Exact Solver by Toro (1999)

Even though no closed-form solutions for the Riemann problem of hydrodynamics exist, it is in fact possible to design an iterative scheme that computes the solution to arbitrary (up to machine) precision. The original idea for such an approach was brought forward by Godunov and Bohachevsky (1959) and was subsequently improved upon by various authors (Chorin 1976, van Leer 1979, among others). Our brief outline shall follow the approach presented in chapter 4 of Toro (1999) and we will assume an equation of state for ideal gases (Eq. 2.11).

The evolution of the Riemann problem in our case equals that of the shock tube problem. As the system evolves, it divides into four distinct states: the initial states (denoted by their primitive vectors) $\vec{W}_\mathrm{L}$ and $\vec{W}_\mathrm{R}$ as well as an intermediate region, often referred to as the *star region*, that itself is again split into a left and right state, $\vec{W}^*_\mathrm{L}$ and $\vec{W}^*_\mathrm{R}$; see Fig. 3.3 for a visualization.

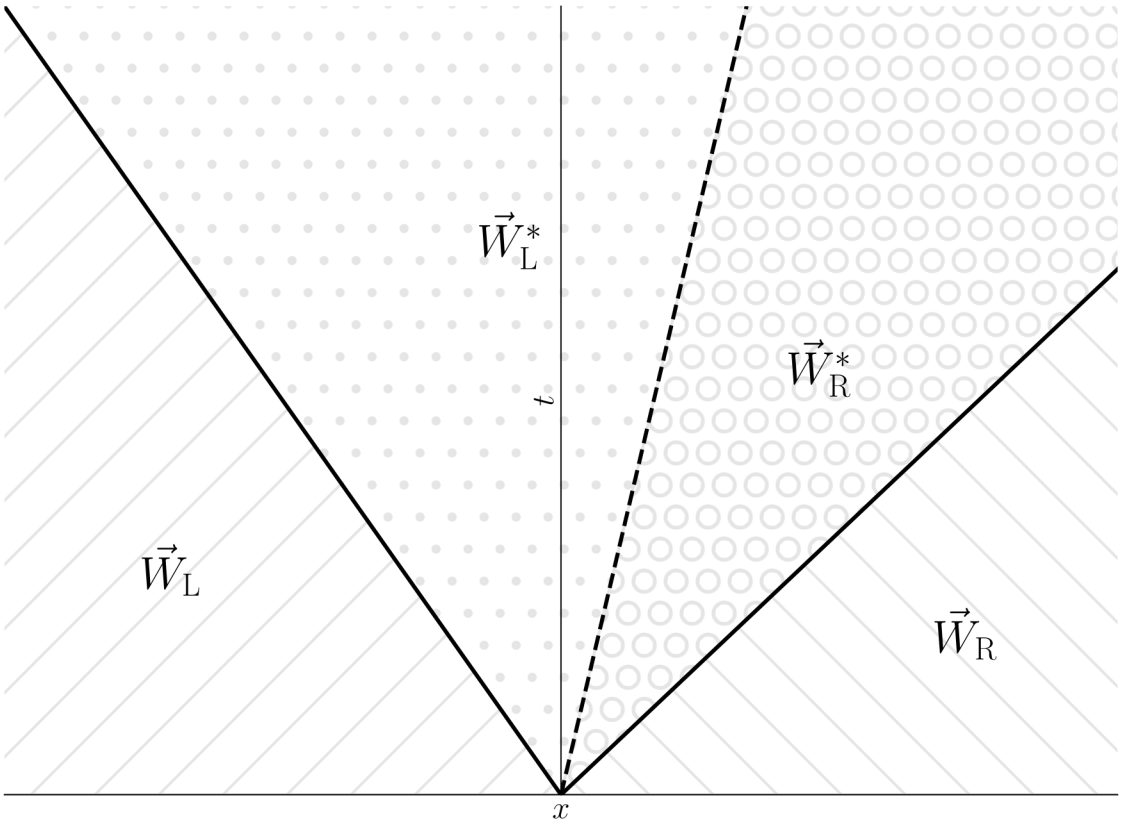


Figure 3.3: The temporal evolution of the Riemann problem. At $\Delta t > 0$, four regions can be identified: the two initial states to the left and right ($\vec{W}_\mathrm{L}$ and $\vec{W}_\mathrm{R}$, respectively) as well as two distinct star regions, $\vec{W}^*_\mathrm{L}$ and $\vec{W}^*_\mathrm{R}$. The direction that the contact discontinuity (dashed line) expands into depends on the initial conditions. The solid lines may either represent rarefaction waves or shock discontinuities.

Theoretical examinations of the Euler equations (as carried out in Toro 1999, chapter

3.1.3) conclude that the pressure p^* and the particle velocity v^*[10] are constant over the entire star region, whereas the density is piecewise constant over each of the two subregions, but in general $\rho^*_\mathrm{L} \neq \rho^*_\mathrm{R}$. The exact solution can then be found by using root-finding algorithms to solve the following equation for p^*:

$$f_\mathrm{L}\left(p^*, \vec{W}_\mathrm{L}\right) + f_\mathrm{R}\left(p^*, \vec{W}_\mathrm{R}\right) + v_\mathrm{R} - v_\mathrm{L} = 0 \ . \tag{3.17}$$

Using the shorthand X to identify either of the two indices L and R, the functions f_X equate to

$$f_X\left(p^*, \vec{W}_X\right) = \begin{cases} (p^* - p_X)\sqrt{\dfrac{A_X}{p^* + B_X}} & \text{if } p^* > p_X \\ \dfrac{2c_{\mathrm{s},X}}{\gamma - 1}\left[\left(\dfrac{p^*}{p_X}\right)^{\dfrac{\gamma-1}{2\gamma}} - 1\right] & \text{if } p^* \leq p_X \end{cases} \tag{3.18}$$

with the speed of sound $c_{\mathrm{s},X}$ and

$$A_X = \frac{2}{(\gamma+1)\,\rho_X} \ , \tag{3.19}$$

$$B_X = \frac{\gamma - 1}{\gamma + 1} p_X \ . \tag{3.20}$$

The velocity of the star region is then easily calculated as

$$v^* = \frac{1}{2}\left(v_\mathrm{L} + v_\mathrm{R}\right) + \frac{1}{2}\left[f_\mathrm{R}\left(p^*, \vec{W}_R\right) - f_\mathrm{L}\left(p^*, \vec{W}_L\right)\right] \tag{3.21}$$

and the densities on both sides within the star region are

$$\rho^*_X = \begin{cases} \rho_X\left(\dfrac{\gamma-1}{\gamma+1} + \dfrac{p^*}{p_X}\right)\left(\dfrac{\gamma-1}{\gamma+1}\dfrac{p^*}{p_X} + 1\right)^{-1} & \text{if } p^* > p_X \\ \rho_X\left(\dfrac{p^*}{p_X}\right)^{\gamma^{-1}} & \text{if } p^* \leq p_X \ . \end{cases} \tag{3.22}$$

Having obtained the full information about the star region, we can now translate its expansion to fluxes and update the particle properties accordingly.

3.3.2 The HLL Solver

While the exact scheme presented above has clear advantages in its physical accuracy, its root-finding algorithm is computationally quite expensive. Taking into account that it will be called for every two-particle interaction, the toll on total runtime can quickly accumulate. Hence, approximate Riemann solvers that strike a balance between required accuracy and performance have been studied. Here, we will present one of the earliest examples of such solvers, developed by Harten et al. (1983).

[10]Since we already reduced the problem to one dimension, we can trivially identify $v = |\vec{v}|$.

Named the HLL solver after the authors of the inaugural publication (Harten, Lax, and van Leer), its ansatz is to calculate fluxes between cells directly, thereby omitting the evaluation of the star region. Furthermore, it is assumed that the system can be characterized by two waves propagating with wave speeds[11] S_L and S_R, separating the left and right states from a single middle state. This explicitly means that the ensuing contact discontinuity is not considered, see Fig. 3.4.

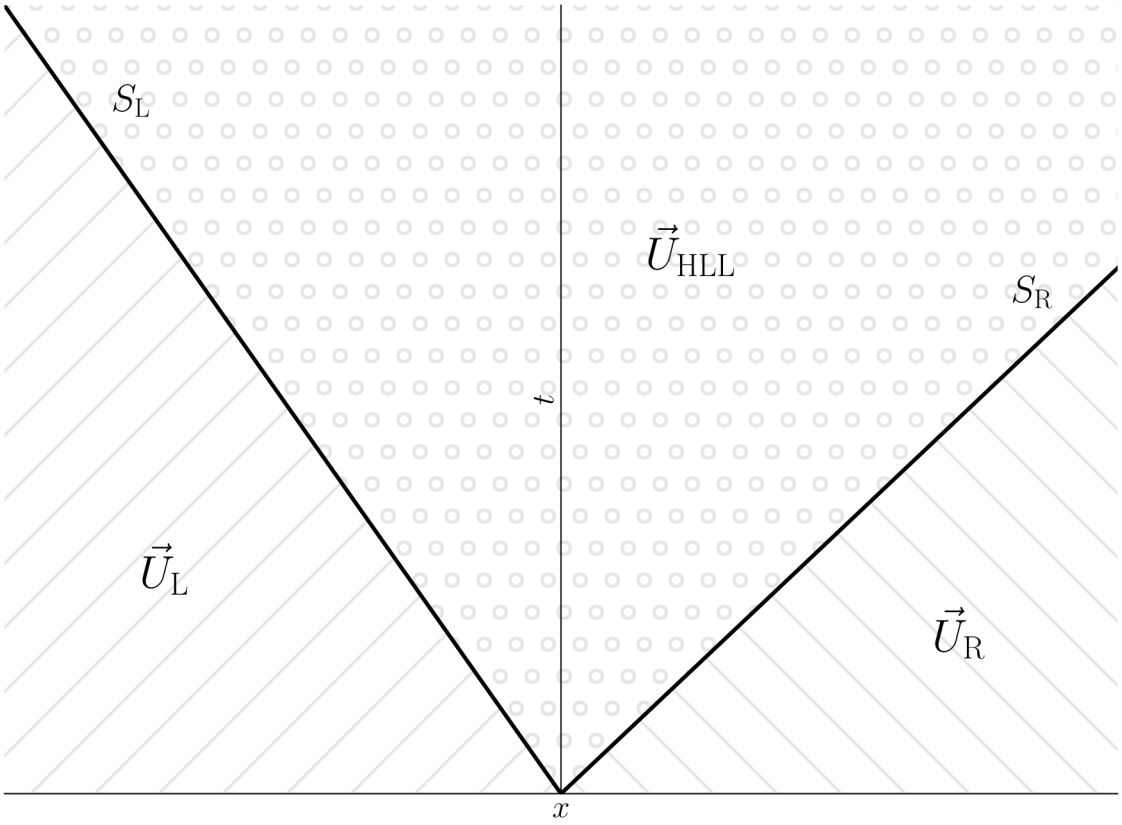


Figure 3.4: The temporal evolution of the Riemann problem in the HLL approach. Contrary to Fig. 3.3, only two waves (with wave speeds S_L and S_R) are considered, separating the initial states from a single intermediate state $\vec{U}_\mathrm{HLL}$.

In a first step, the wave speeds S_X must be estimated. The simplest approach is to infer them from a particle's gas and sound speed as

$$S_X = v_X - c_{\mathrm{s},X} \ . \tag{3.23}$$

With the wave speeds obtained, we can now directly compute the intercell fluxes as

[11] Keeping in mind that we are dealing with ensembles of gas particles within each simulated particle, a more precise formulation would be: the fastest signal velocities perturbing the initial data (Toro, 1999).

$$\vec{F}_{\mathrm{HLL}} = \begin{cases} \vec{F}\left(\vec{U}_{\mathrm{L}}\right) & \text{if } 0 \leq S_{\mathrm{L}} \\ \dfrac{S_{\mathrm{R}}\vec{F}\left(\vec{U}_{\mathrm{L}}\right) - S_{\mathrm{L}}\vec{F}\left(\vec{U}_{\mathrm{R}}\right) + S_{\mathrm{L}}S_{\mathrm{R}}\left(\vec{U}_{\mathrm{R}} - \vec{U}_{\mathrm{L}}\right)}{S_{\mathrm{R}} - S_{\mathrm{L}}} & \text{if } S_{\mathrm{L}} \leq 0 \leq S_{\mathrm{R}} \\ \vec{F}\left(\vec{U}_{\mathrm{R}}\right) & \text{if } S_{\mathrm{R}} \leq 0 \end{cases} \tag{3.24}$$

where the $\vec{F}(\vec{U}_X)$ are evaluated using the definitions of $\vec{F}$ (Eq. 3.2) and $\vec{U}$ (Eq. 2.44) in combination with the equation of state. For a full derivation of Eq. (3.24), we recommend the lecture of chapter 10 in Toro (1999).

The HLL solver is fairly simple to implement and, in most cases, produces adequate results. Nevertheless, due to its negligence of contact discontinuities, it struggles in cases where these become important on macroscopic scales as well as in the presence of shear waves (which produce similar intermediate waves). To overcome these issues, several advancements of the simple HLL solver have been proposed, such as the HLLE solver (Einfeldt 1988, see also Davis 1988 for a similar ansatz) that uses a more educated guess for the wave speeds S_X or the HLLC solver (Toro et al., 1994) which assumes a three-wave model.

3.4 Flavours: MFV and MFM

Although it might seem that we have defined the meshless approach rather rigorously so far, there is still quite some amount of customization left that we can utilize to enforce desired behaviours.

For one, whenever we previously referred to a velocity $\vec{v}$ in this section, we implicitly meant the fluid velocity. There is no need per se for this velocity to always equal the velocity of our (Lagrangian) particles. We could, for example, keep the particle velocities to be exactly zero at all times in which case our meshless method would become fully Eulerian. We will, however, assume that each particle i indeed moves with the fluid velocity at $\vec{r}_i$ to obtain a wholly Lagrangian description. We stress that these two choices merely represent two extrema in a wide field of possibilities that is as of yet only very sparsely explored.

Secondly, when integrating the fluxes over the time step Δt, we should take into account that the face between a pair of particles will not remain at a fixed location over time. This is even true after we rotated our reference frame so as to reduce the problem to one spatial dimension. Nevertheless, one realization of the meshless method is to simply neglect this effect (since we did boost to the reference frame of the quadrature point, the corrections are in most cases small enough to justify this). This has the advantage of preserving the volume of both cells in each interaction, and therefore the volume of every cell. Hence, this method is called *meshless finite volume* (MFV), adopting the naming convention of Hopkins (2015).

Another way to estimate the motion of the face is to associate it with the motion of the star region within the Riemann problem (see Fig. 3.3 for a visualization and Eq. 3.21 for an exact solution thereof). Effectively, we are moving the cell face with the velocity of

the contact discontinuity[12] with mass being conserved on either side. Consequently, this manifestation is called *meshless finite mass* (MFM).

3.5 Time Steps and Time Integration

In principle, the scheme for determining the time steps Δt for each two-particle interaction is not inherently dictated by the meshless scheme and can be chosen independently, as long as consistency is ensured. As is usual in CFD, however, the goal is to resolve highly dynamical environments more finely (i.e. with smaller Δt) than quiescent ones. Here, we will exemplarily outline the approach of Hopkins (2015) which itself is derived from Springel (2010).

Time steps are assigned to each particle based on a Courant-Friedrichs-Lewy criterion, ensuring that information cannot travel further than one cell during each step. Following Whitehurst (1995) and Monaghan (1997), the time steps are calculated as

$$(\Delta t)_{\mathrm{CFL},i} = 2C_{\mathrm{CFL}} \frac{h_i}{\left|v_{\mathrm{sig},i}\right|} \tag{3.25}$$

with the parametric Courant-Friedrichs-Lewy number $C_{\mathrm{CFL}} \leq 1$ dictating what fraction of a kernel with diameter $2h$ can be crossed per time step and the signal velocity

$$v_{\mathrm{sig},i} = \max_j \left\{ c_{\mathrm{s},i} + c_{\mathrm{s},j} - \min \left[0, \frac{\left(\vec{v}_i - \vec{v}_j\right)\left(\vec{r}_i - \vec{r}_j\right)}{\left|\vec{r}_i - \vec{r}_j\right|} \right] \right\} \tag{3.26}$$

determined over all neighbours j for particle i. The obtained time steps are afterwards arranged in a power-of-two hierarchy by reducing each $(\Delta t)_{\mathrm{CFL},i}$ to the next smallest available time bin in order to facilitate synchronization. This procedure is coupled with a so-called wake up mechanism (see Saitoh and Makino 2009) which limits the maximum difference in assigned time steps in adjacent particles. If a violation is detected, the time step of the particle in the higher time bin is reduced accordingly.

Having assigned the correct time steps, we can then update our conservative variables (via the proxy $\vec{Q}_i \equiv (V\vec{U})_i$) from one time step n to the next step $n+1$ as

$$\vec{Q}_i^{(n+1)} = \vec{Q}_i^{(n)} - (\Delta t)_i \sum_j \vec{A}_{ij} \vec{F}_{ij}^{n+1/2} \tag{3.27}$$

with the time-centred flux $\vec{F}_{ij}^{n+1/2}$ calculated from the predicted primitive vectors at time step $n+1/2$ (see Eq. 3.16). As soon as one particle's time step becomes active, fluxes are calculated and the conserved quantities are updated for both particles. Primitive variables, however, are only updated for active particles.

3.6 First Results

The meshless approach is, still, in an early stage and lacks the plethora of research and expertise that the methods we discussed in Sec. 2 can draw from. However, it has since

[12] It should be stressed that we rely heavily on the simplification of the Riemann problem to one dimension. In violent environments in simulations in two or three dimensions, we can thus not expect to capture the whole complexity and our estimates for the velocity of the star region will (even when using an exact solver) become somewhat incorrect.

its inauguration been able to produce very promising results and we shall highlight some of the work already done here.

Already in the very first application of a meshless scheme in an astrophysical context, Gaburov and Nitadori (2011) implemented a full magneto-hydrodynamical (MHD) treatment. As is illustrated in Fig. 3.5, their MFV-scheme was able to maintain the constraint $\nabla \vec{B} \stackrel{!}{=} 0$ reasonably well even on long timescales.

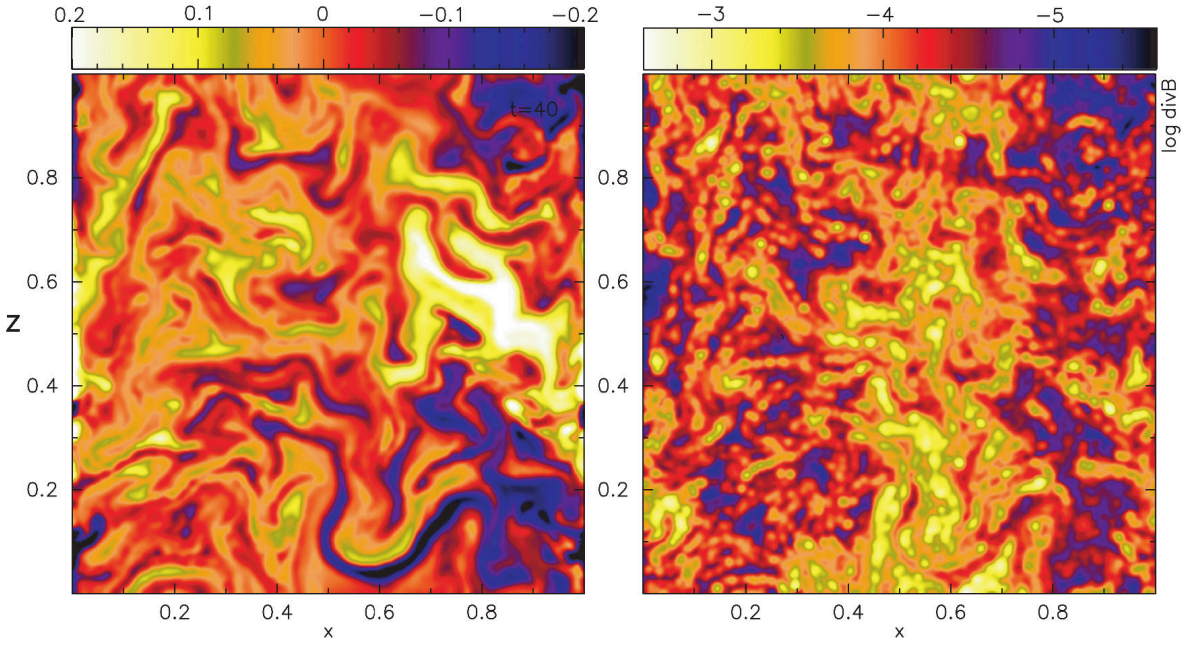


Figure 3.5: 2d axisymmetric shearing box with magnetic field strength $\vec{B}$ (left) and $\nabla \vec{B}$ (right) after considerable simulation time; taken from Gaburov and Nitadori (2011).

Hopkins (2015) carried out a comprehensive comparison between the SPH, meshless, and Eulerian methods in the code `GIZMO`. Their conclusion is that both MFM and MFV were at least on par with established SPH, AMR, or moving mesh methods. In all cases, the conservation of angular momentum was fulfilled more accurately. Some of the finer findings of their work include:

- Compared to SPH solvers, meshless schemes are less diffusive and have reduced numerical viscosity. The smaller number of required neighbours lessens the effect of over-smoothing of discontinuities. In addition, instabilities and fluid mixing are resolved better without requiring additional corrective terms, see Fig. 3.6.
- Compared to AMR methods, MFM and MFV were free of grid alignment phenomena and did not produce spurious grid heating when coupled to gravity. In addition, advection errors could be minimized.
- While the differences to moving mesh methods were much more subtle, mesh deformation noise could be slightly reduced.
- Differences between the two meshless schemes include better angular momentum conservation and slightly reduced noise when choosing MFM over MFV. On the other hand, the latter is less diffusive and spreads contact discontinuities over a smaller fraction of the kernel width. This resolves phase boundaries more sharply, as can be seen in Fig. 3.6.

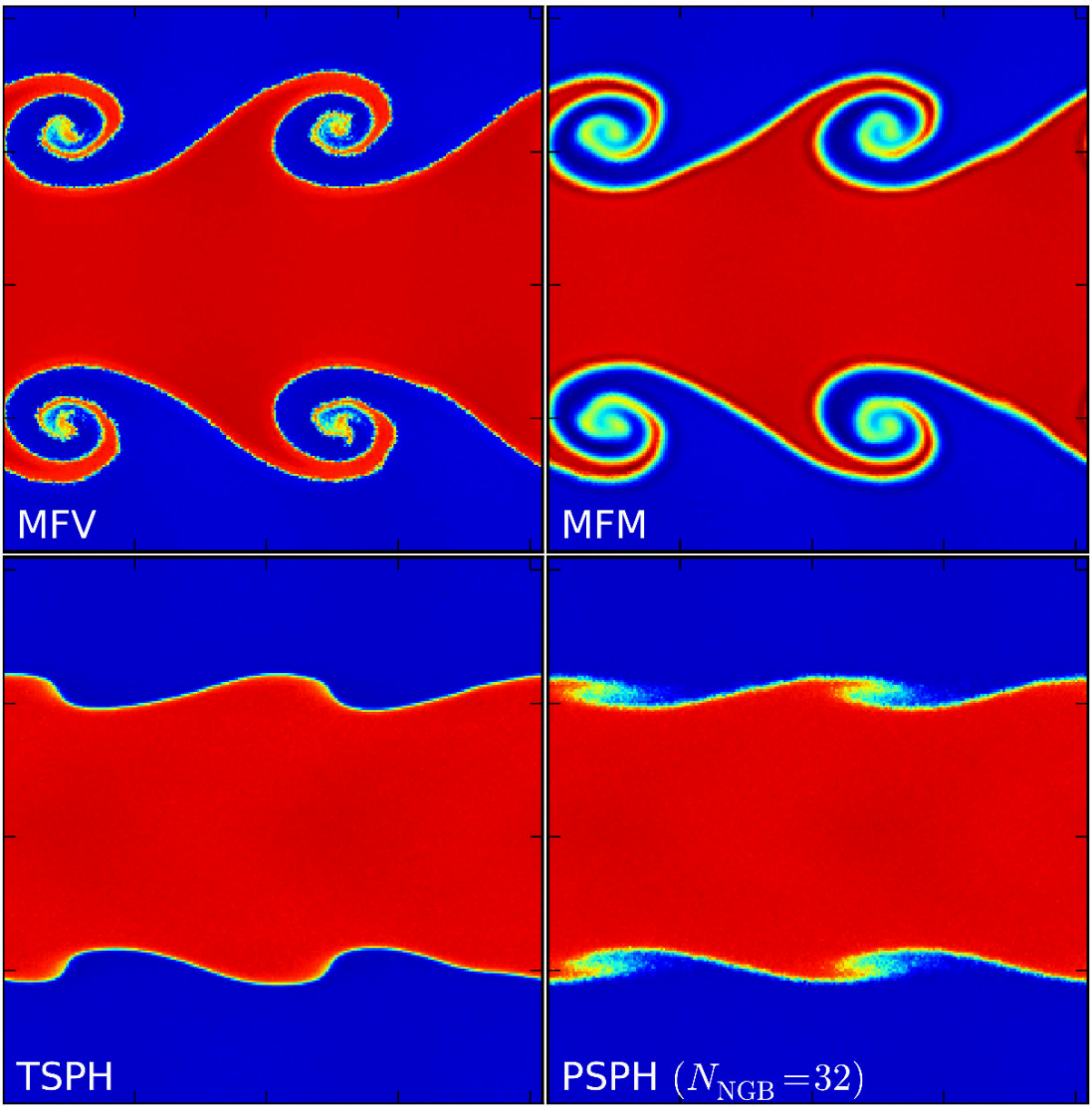


Figure 3.6: Comparison of the Kelvin-Helmholtz instability using meshless schemes (top panels) and two different formulations of SPH (bottom panels). *TSPH* refers to the classical approach as in `GADGET` (Springel, 2005) whereas *PSPH* refers to an alternative formulation as in Hopkins (2013). The number of neighbours is kept constant across all simulations. Figure taken from Hopkins (2015).

Recent astrophysical results are presented by Davé et al. (2019) who carried out the cosmological galaxy formation simulation suit `SIMBA` using `GIZMO`, including stellar and AGN feedback, radiative cooling, photoionisation, an H_2 based star formation rate, chemical enrichment, metal-loaded galactic winds, dust, and a black hole growth model, see Fig. 3.7. The authors opted to use MFM over MFV for its reduced noise and easier coupling to additional physics. They were able to reproduce numerous observables such as galaxy stellar mass functions, the stellar mass-star formation rate main sequence, sizes of star-forming galaxy, and hot gas fractions in massive halos. However, their inferred mass function at $z = 0$ features an insufficiently sharp truncation and low-mass quenched galaxies are too large in size.

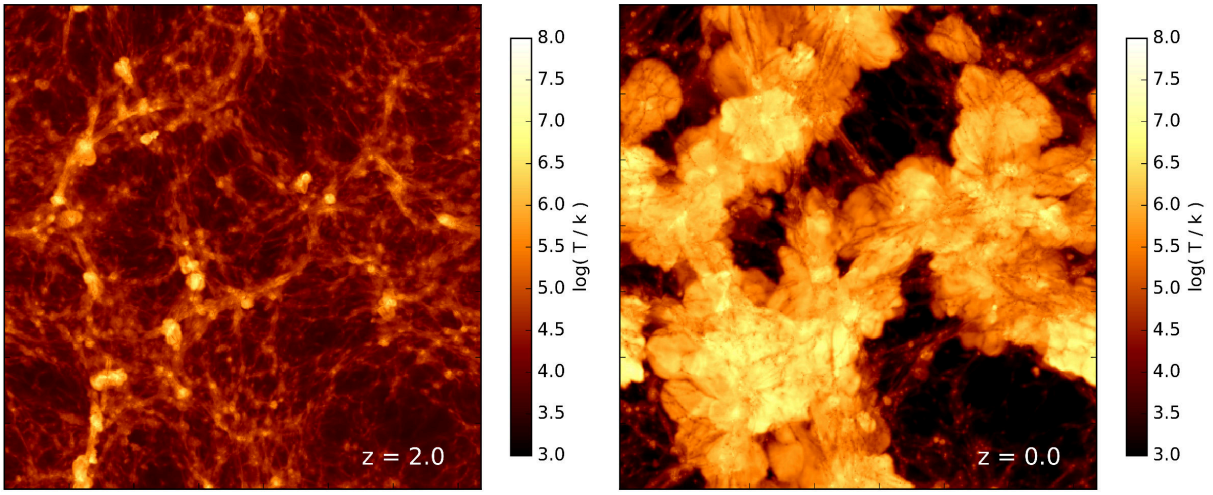


Figure 3.7: Randomly selected $10\,\mathrm{Mpc\,h^{-1}}$ slice of a $50\,\mathrm{Mpc\,h^{-1}}$ `SIMBA` cube at redshifts $z = 2$ and $z = 0$; taken from Davé et al. (2019).

4 The GADGET Code

`GADGET` is an N-body/SPH code designated to cosmological studies, both of large-scale structures and of individual objects such as galaxies and clusters. It is written in the `C` programming language and focuses heavily on parallel execution on supercomputers. `GADGET` was originally authored and published by Volker Springel (Springel et al., 2001) as a means to study **ga**laxies with **d**ark matter and **g**as int**e**rac**t**ion. Over the years, many contributors have advanced the scope, capabilities and performance of the initial code a great deal, see e.g. Springel (2005), Dolag and Stasyszyn (2009), Petkova and Springel (2009), Arth et al. (2014), and Beck et al. (2016).

Currently, `GADGET-2` is freely available at `https://wwwmpa.mpa-garching.mpg.de/gadget/`. Due to the open source nature of the code and its popularity, many derivatives exist concurrently, both open source and unpublished versions. The iteration that will be described in this chapter and which was the focus of this work is `OpenGadget3`, an as of yet unpublished derivative whose development is led by Klaus Dolag at Max Planck Institute for Astrophysics and the university observatory of Ludwig-Maximilians-Universität Munich as well as by Stefano Borgani at INAF observatory Trieste.

This section shall highlight the key aspects of `OpenGadget3` (many of which shared with other `GADGET` iterations) and give a non-exhaustive overview of the scientific results that were achieved with it. We will focus on the treatment of physics and ignore aspects such as code structure and parallelisation techniques.

4.1 The Gravity Solver

The details of the treatment of gravity in `OpenGadget3` are explicated in Springel (2005). We will limit ourselves here to a drastically simplified description.

The general approach used to solve gravitational interactions is that of an N-body solver: Phase space is populated by a discrete number N of tracer particles and interactions are calculated between them. The Hamiltonian of the ensemble in Newtonian space is then given by

$$H = \sum_{i=1}^{N} \frac{\vec{p}_i^{\,2}}{2m_i} + \frac{1}{2} \sum_{i=1}^{N} \sum_{\substack{j=1 \\ i \neq j}}^{N} m_i m_j \varphi\big(\vec{r}_i - \vec{r}_j\big) \tag{4.1}$$

with the particle momentum $\vec{p}_i$ and the interaction potential

$$\varphi\left(\vec{r}_i - \vec{r}_j\right) = \frac{-G}{\left|\vec{r}_i - \vec{r}_j\right|} \ , \tag{4.2}$$

G being Newton's constant. Using

$$\frac{\mathrm{d}\vec{p}_i}{\mathrm{d}t} = -m_i \sum_j \frac{G m_j}{\left|\vec{r}_i - \vec{r}_j\right|^3} \left(\vec{r}_i - \vec{r}_j\right) \tag{4.3}$$

and

$$\frac{\mathrm{d}\vec{r}_i}{\mathrm{d}t} = \frac{\vec{p}_i}{m_i} \tag{4.4}$$

one can then use simple numerical integration schemes to evolve the system indefinitely[13].

[13] Although some precautions should be taken concerning the order in which Eqs. (4.3) and (4.4) are solved, see Sec. 4.3.

This fundamental ansatz is improved upon in GADGET in a threefold manner to make the simulation both less time consuming and more physically accurate.

1. Since every (simulated) particle actually represents a whole ensemble of (physical) particles (such as stars), gravity should be realized in a collisionless manner. We can ensure this by spreading the mass m_i associated with each particle within a finite region around it, thus avoiding the singularity at $|\vec{r}_j - \vec{r}_j| \to 0$. This mass smoothing is done using a kernel function akin to those we discussed in Sec. 2.2.3. To avoid confusion, we denote the gravitational smoothing length, also referred to as the softening length, with ϵ and reserve h for fluid smoothing. Generally, ϵ is constant for all particles but a scheme for adaptive gravitational softening in GADGET is available as well (Iannuzzi and Dolag, 2011).

2. Eq. (4.1) prominently features a double sum in the second term. Since both of these sums run over all N particles, the execution time using a direct evaluation would scale with N^2, making resolution improvements very costly. To avoid this behaviour, GADGET refrains from calculating gravitational interactions on a strict particle-to-particle basis while still maintaining the long-range nature of the force. This is realized with the implementation of a so-called tree code (see Barnes and Hut 1986). To this end, the computational domain is split into a hierarchical set of subdivisions. Starting from a cubical *root* node comprising the entire set of particles, subdivisions are performed by dividing each node into 2^d equally sized daughter nodes (with dimensionality d). This process concludes once every node only contains a single particle. These nodes are then referred to as *leaf* nodes. Upon calculating the gravitational forces, particles are replaced by their respective nodes with the explicit possibility of grouping them together. The second sum in Eq. (4.1) follows this hierarchical structure of nodes (the *tree*) from root to leaf, in each step checking if a node (encompassing more than one particle) can enter the sum as a whole or if a smaller subdivision must be considered. A node is refined if the following criterion is met:

$$\frac{GM}{r^2}\left(\frac{l}{r}\right)^2 > \alpha\,|\vec{a}| \tag{4.5}$$

   with the total mass inside the node M, its distance from the particle being evaluated (particle i) r, its extension l, the size of the total acceleration obtained in the last time step $\vec{a}$, and a tolerance parameter α. Evidently, the distance of a node to the particle in question is the most important factor, followed by its spatial extension and only then by its mass. Since the obtained result is naturally only an approximation of the exact result, α can be used to tune the accuracy of which to an arbitrary degree. Using this approach, the scaling is reduced from N^2 to $N \log N$.

3. In full cosmological runs, cosmic expansion is accounted for. See Sec. 4.4 for details.

4.2 The Hydro Solver

Since gravity is handled as an N-body problem (i.e. on a particle basis), it was a natural decision to apply a Lagrangian method to solve hydrodynamical interactions in GADGET. More specifically, SPH (see Sec. 2.2) was chosen due to its established performance, exact conservation properties, and the amount of research already available regarding this approach.

The default kernel in `OpenGadget3` is the M^4 kernel (see Eq. 2.23) although, based on the findings of Dehnen and Aly (2012), either the C^6 kernel (Eqs. 2.24 and 2.25) or the similar C^4 kernel are now used predominantly for their improved accuracy and stability.

The basic SPH method is improved upon by a number of additions such as artificial viscosity, a shear flow limiting procedure, and artificial conduction. See Beck et al. (2016) for an overview of the current state of `OpenGadget3`'s hydrosolver.

4.3 Time Stepping

`OpenGadget3` evolves its particles using the kick-drift-kick (KDK) mode of a leapfrog integrator. A *kick* refers to an update of the particles momenta whereas a *drift* calculates the change in position of a particle. These are done in turn and synchronization is regained after each full KDK step. In practice, this means that evolving a single particle i from a distinct time step n to the next step $n+1$ with a given $(\Delta t)_i$ is achieved as

$$\text{First half-kick:} \quad \vec{p}_i^{\,n+1/2} = \vec{p}_i^{\,n} + \vec{f}_i^{\,n-1/2}\frac{(\Delta t)_i}{2} \tag{4.6}$$

$$\text{Drift:} \quad \vec{r}_i^{\,n+1} = \vec{r}_i^{\,n} + \frac{\vec{p}_i^{\,n+1/2}}{m_i}(\Delta t)_i \tag{4.7}$$

$$\vec{f}_i^{\,n-1/2} \mapsto \vec{f}_i^{\,n+1/2} \tag{4.8}$$

$$\text{Second half-kick:} \quad \vec{p}_i^{\,n+1} = \vec{p}_i^{\,n+1/2} + \vec{f}_i^{\,n+1/2}\frac{(\Delta t)_i}{2} \tag{4.9}$$

with $\vec{f}_i$ being the sum of all forces acting on particle i. The exact nature of the force update (Eq. 4.8) depends on the physics applied for each simulation and may become arbitrarily complex.

In addition, `OpenGadget3` allows for individual time steps for each particle. Similar to the time stepping approach we discussed for meshless schemes (Sec. 3.5), the calculated time steps $(\Delta t)_{\text{calc}}$ are then rearranged in a power-of-two hierarchy of refined time steps. This ensures that at the end of each maximum time step $(\Delta t)_{\text{max}}$, all particles are manifestly synchronized. The code advances time in instances of the smallest occupied time bin and every particle is drifted during each time step (Eq. 4.7). The kicks (Eqs. 4.6 and 4.9), however, as well as the force update will only be calculated for those particles that have reached the end of their time step, i.e. *active* particles.

The formal calculation of the particle time steps $(\Delta t)_{\text{calc}}$ for SPH (i.e. gas) particles follows Eq. (3.25). For gravitational particles, the time step is calculated as

$$(\Delta t)_{\text{grav},i} = \sqrt{\frac{2\eta\epsilon}{|\vec{a}_i|}} \tag{4.10}$$

with the gravitational softening ϵ, acceleration $\vec{a}_i = \vec{f}_i/m_i$, and an accuracy parameter η. Additional physics such as artificial conduction, stellar feedback, and black hole inclusion may impose different time step constraints on affected particles. The time step that is used for placing a particle into the hierarchical time bin structure always equals the minimum of all applicable time steps.

4.4 Cosmology

Simulations are not limited to Newtonian space but can also be carried out with a cosmological background. In theory, `OpenGadget3` can be run with arbitrary cosmologies so long as they unambiguously produce a scale factor $a(t)$[14].

The scale factor, quantifying the expansion (or, theoretically, contraction) of space, is taken into account when evaluating the central equations laid out so far. Most importantly, the Hamiltonian becomes

$$H = \sum_i \frac{\vec{p}_i^{\,2}}{2m_i a(t)^2} + \frac{1}{2}\sum_i \sum_j \frac{m_i m_j \varphi\big(\vec{r}_i - \vec{r}_j\big)}{a(t)} \tag{4.11}$$

and the KDK scheme (where now t corresponds to the total time of time step n) is modified as

$$\text{First half-kick:} \quad \vec{p}_i^{\,n+1/2} = \vec{p}_i^{\,n} + \vec{f}_i^{\,n-1/2} \int_t^{t+(\Delta t)_i/2} \frac{\mathrm{d}t'}{a(t')} \tag{4.12}$$

$$\text{Drift:} \quad \vec{r}_i^{\,n+1} = \vec{r}_i^{\,n} + \frac{\vec{p}_i^{\,n+1/2}}{m_i} \int_t^{t+(\Delta t)_i} \frac{\mathrm{d}t'}{a(t')^2} \tag{4.13}$$

$$\vec{f}_i^{\,n-1/2} \mapsto \vec{f}_i^{\,n+1/2} \tag{4.14}$$

$$\text{Second half-kick:} \quad \vec{p}_i^{\,n+1} = \vec{p}_i^{\,n+1/2} + \vec{f}_i^{\,n+1/2} \int_{t+(\Delta t)_i/2}^{t+(\Delta t)_i} \frac{\mathrm{d}t'}{a(t')} \,. \tag{4.15}$$

Furthermore, the evaluations are performed using comoving coordinates $\vec{r}_\mathrm{c} = \vec{r}/a(t)$ to ensure stable resolution for the entirety of the simulation.

4.5 Magnetic Fields

Dolag and Stasyszyn (2009) complemented `GADGET` with a comprehensive treatment of magnetic fields. Their implementation makes use of the points we presented in Sec. 2.2.4 and couples them with various stabilization and regularization mechanisms:

- In order to suppress the clumping instability (Phillips and Monaghan, 1985), a correction term to the magnetic force was introduced, following Børve et al. (2001). This term explicitly subtracts the effects of any non-vanishing ∇B contributions and is calculated as

$$\left(\frac{\mathrm{d}\vec{v}_i}{\mathrm{d}t}\right)_\mathrm{corr} = -\frac{\mathrm{d}t}{\mathrm{d}\eta}\frac{\beta}{\mu_0}\vec{B}_i \sum_j m_j \left[f_i \frac{\vec{B}_i}{\rho_i^2} \cdot \nabla_i W\big(\vec{r}_i - \vec{r}_j, h_i\big) + f_j \frac{\vec{B}_j}{\rho_j^2} \cdot \nabla_j W\big(\vec{r}_i - \vec{r}_j, h_j\big) \right] \tag{4.16}$$

  with the conformal time η, a control parameter $\beta \leq 1$ and f_i given by Eq. (2.34).

- To reduce numerical noise within the magnetic fields (especially on scales smaller than typical smoothing lengths), the magnetic field itself is treated as a smoothed quantity in accordance to Eq. (2.14).

- To the same extend, artificial magnetic dissipation was introduced, analogous to the implementation of artificial viscosity we discussed in Sec. 2.2.5. Based on the findings of Price and Monaghan (2004), this dissipation term scales with the change of the total magnetic field.

[14] A Newtonian universe, in actuality, is nothing else than the edge case $a(t) \equiv 1$.

4.6 Sub-Grid Physics

A general drawback of every cosmological code is that in order to have sufficient statistics on large scales, resolution must be chosen rather coarsely. A single particle (or cell) may easily contain thousands or even millions of $M_\odot$, making any simulation of internal or external processes for single objects unfeasible by orders of magnitude. However, some of these processes have important consequences even on scales that are large enough to be resolved. In the following, some of these processes and their approximative treatment in `OpenGadget3` shall be highlighted.

4.6.1 Star Formation and Stellar Feedback

The standard model for star formation in `GADGET` was introduced by Springel and Hernquist (2003). In it, hot and cold gas coexist in each gas particle in pressure equilibrium. The equations of hydrodynamics are solved solely for the hot gas which is assumed to encase cold clouds that fuel star formation. Combined with the emerging stellar particles, this results in three different phases that baryons can occupy with the following processes linking these:

- Gas may be transferred from the hot to the cold phase by radiative cooling (see e.g. Tornatore et al. 2003).
- Cold clouds may form star particles[15]. Due to resolution constraints, these always represent a whole population of stars, taken to be drawn from a predetermined initial mass function (IMF). IMFs that can be chosen include those brought forward by Salpeter (1955), Kroupa (2001), and Chabrier (2003) but the implementation of custom IMFs is very straightforward. The star formation rate follows a Kennicutt-Schmidt relation (Schmidt 1959 and Kennicutt 1989).
- Massive stars explode as supernovae. This recycles mass and energy to the hot phase while cold clouds are partially evaporated.

A similar approach of modelling star formation and multiphase particles in `GADGET` is presented in Murante et al. (2010). In their multiphase particle integrator (`MUPPI`) model, star formation efficiency is based on the dynamical time of the cold phase, thus not imposing an ad hoc Kennicutt-Schmidt relation. Their model is, however, able to reproduce such. Additionally, the interactions between all three baryon phases are described using a number of ordinary differential equations that are integrated for each gas particle, thereby abandoning the reliance on equilibrium solutions. While `MUPPI` is not yet part of the standard repository of `OpenGadget3`, it is still in active development (see e.g. Valentini et al. 2020) and intermediate results look promising.

4.6.2 Chemical Evolution

The chemical enrichment model follows Tornatore et al. (2007). Taking into account the mass-lifetime relation of stars produced from the IMF (as per Padovani and Matteucci 1993) allows for metal releases from supernovae Type II (Woosley and Weaver, 1995) and Type Ia (Thielemann et al., 2003) as well as those of asymptotic giant branch stars (van den Hoek and Groenewegen, 1997) to be tracked. Lifetime functions, stellar yields, and

[15]This occurs as soon as a gas particle exceeds a certain temperature threshold $T_{\rm SFR}$ to trigger thermal instabilities as well as a density threshold depending on $T_{\rm SFR}$, the characteristic star formation time scale, and the energy budget from supernovae.

the IMF itself are adjustable. This scheme poses another time step constraint on star particles to ensure that a constant percentage of supernovae of each type explode during each Δt.

4.6.3 Cosmic Rays

A more recent addition to `OpenGadget3` is the inclusion of cosmic rays. Following the work of Miniati (2001), Böss et al. (in prep.) implemented a scheme that simulates both protons and electrons as populations contained within gas particles. Their treatment takes into account source terms such as shock injection and supernovae, adiabatic changes due to the excitation of Alfvén waves (see e.g. Lerche 1967 for theoretical background), radiative changes, and turbulent reacceleration.

Although extensive tests on the effects of this inclusion are yet to be carried out, cosmic rays are expected to be of particular importance in galaxies due to various interaction processes with the interstellar medium, see e.g. Grenier et al. (2015) for a comprehensive review.

4.6.4 Black Hole Growth Model and AGN Feedback

Black holes (BHs) in `GADGET` are realized as an own subspecies of particles with the ability to accumulate mass either from accreted gas or by merging with other BHs. Their exact growth model and feedback as active galactic nuclei (AGNs) follow Springel et al. (2005a) with modifications by Fabjan et al. (2010) and Hirschmann et al. (2014).

BHs are seeded inside a halo of dark matter particles whenever its stellar mass component reaches a certain threshold ($\sim 10^{10}\,\mathrm{M}_\odot\,\mathrm{h}^{-1}$). The determination which particles constitute a halo is done with a friends-of-friends algorithm, grouping particles based on proximity. Once formed, BHs may be pinned to their host halo (by forcing them to the position of the particle with the minimum potential within the BH's kernel) or be allowed to move freely. The latter avoids spurious migration of BHs from satellite galaxies to the central halo galaxy but additional measures[16] have to be taken to keep BHs in the potential minimum (i.e. the centre of their host galaxy).

BH mergers are implemented in a straightforward manner, combining the masses of the two progenitors into a single particle when their distance drops below a specified threshold. The accretion rate onto a BH is estimated using the Bondi-Hoyle-Littleton approximation (see e.g. Bondi 1952) as

$$\dot{M}_\bullet = \frac{4\pi G^2 M_\bullet^2 \alpha \rho}{\left(c_\mathrm{s}^2 + v_\bullet^2\right)^{3/2}} \tag{4.17}$$

with ρ and c_s being the density and sound speed, respectively, of the surrounding gas, $v_\bullet$ the velocity of the BH with respect to the surrounding gas, and α a boost factor accounting for the fact that gas properties in the near vicinity of the BH cannot be resolved accurately. Gas particles may either transfer all of their mass or only a (predetermined) fraction of it onto the BH. Additionally, accretion rates can never exceed the Eddington limit given by the necessary outwards directed radiation pressure needed to balance out the inwards directed gravitational forces.

[16] These measures include a strict conservation of momentum for gas accretion as well as momentum and centre of mass conservation for BH mergers. Additionally, an aggressive gravitational softening is applied.

AGN feedback is directly proportional to the accretion rate onto the black hole and realized as thermal energy being injected to the surrounding gas particles according to their kernel weight. The rate of this feedback is given by

$$\dot{E}_{\mathrm{AGN}} = \epsilon_{\mathrm{r}} \epsilon_{\mathrm{f}} \dot{M}_{\bullet} c^2 \tag{4.18}$$

with the speed of light c, $\epsilon_{\mathrm{r}} \approx 0.1$ being the radiative efficiency and ϵ_{f} a free parameter. ϵ_{f} can be used to distinguish between different environments, e.g. in quasars $\epsilon_{\mathrm{f}} = 0.15$ whereas radio-mode feedback would constitute $\epsilon_{\mathrm{f}} = 0.6$ (see Hirschmann et al. 2014). Lastly, the accretion rate is adjusted for this feedback so that

$$\Delta M_{\bullet} = (1 - \epsilon_{\mathrm{r}})\, \dot{M}_{\bullet} \Delta t \ . \tag{4.19}$$

4.7 Scientific Results

The scientific goal of `GADGET` and all its iterations is to study structure formation and evolution on large scales. Hence, the majority of research carried out using this code is concerned with galaxies, clusters, and large-scale structure itself.

Querying NASA's Astrophysics Data System (ADS) for articles citing Springel (2005) (the article accompanying the release of `GADGET-2`) alone yields thousands of results and discussing them all would surely exceed the scope of this (or any) work. We will therefore limit ourselves here to a very short and subjective selection of results that were achieved with different versions of the code.

One of the earliest astrophysical investigations using `GADGET` was published by Borgani et al. (2004). Assuming a ΛCDM cosmology, the authors studied the X-ray properties of clusters and groups of galaxies, using a cosmological box with a side length of $192\,\mathrm{Mpc\,h^{-1}}$ as well as 480^3 gas particles and as many dark matter particles. Mass-temperature relations, X-ray temperature functions and luminosity-temperature relations were all found to be in good agreement with observations. However, their simulated clusters suffered from overcooling and thus, the fraction of baryons in stars was overestimated.

The capabilities of `GADGET-2` were impressively demonstrated with the execution of the `Millennium` run (Springel et al., 2005b), simulating $2{,}160^3$ dark matter particles in a box of side length 2.23 Gly and following their evolution from $z = 127$ to $z = 0$. See Fig. 4.1 for a depiction of the dark matter component at $z = 0$. Galaxies and quasars were added in post-processing. The authors were able to strengthen the idea that small baryon-induced perturbations in the initial conditions would propagate to become features now observable in the distribution of low-redshift galaxies.

Kereš et al. (2009) studied avenues of galaxy formation using a $50\,\mathrm{Mpc\,h^{-1}}$ box with 288^3 particles of dark and baryonic matter each. They found that, contrary to many historic analytical models, cooling of shock-heated virialized gas only plays a minor role whereas most mass is accreted through inflow from filaments, independent of the mass of the galaxy.

More recently, the non-public version `GADGET-3` was used to simulate a suite of cosmological boxes with various resolutions, named the `Magneticum Pathfinder` simulations (see Hirschmann et al. 2014 and Ragagnin et al. 2017, also see Fig. 4.2). Various studies have since been carried out using the (in part publicly available) data for scientific investigations. Teklu et al. (2015) conducted an extensive study connecting galaxy morphologies to their angular momentum. They found that simulations agreed well with observations

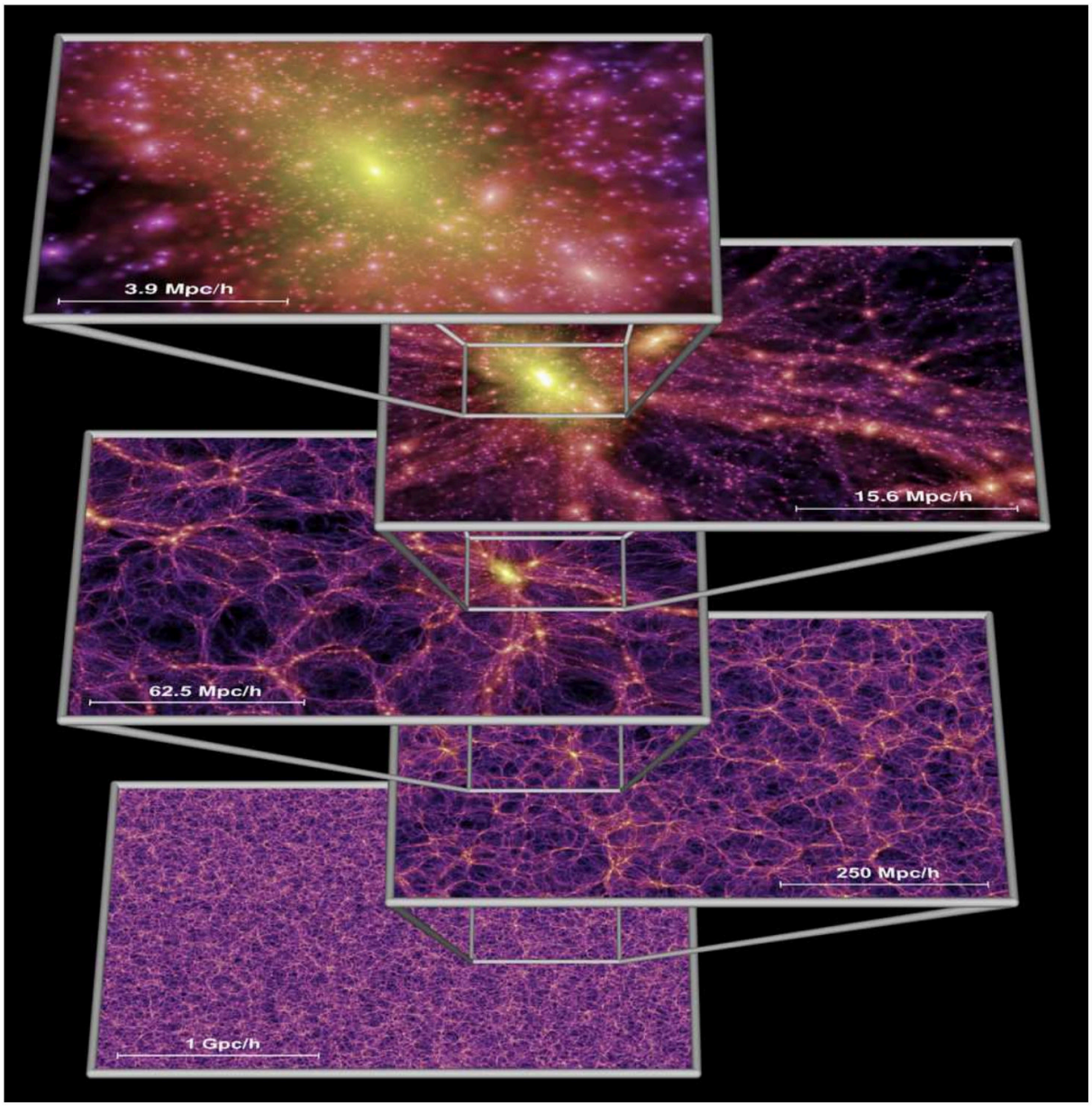


Figure 4.1: The dark matter distribution in the `Millennium` simulation at $z = 0$ in various zoom levels. The colour code represents density and local dark matter velocity dispersion. Figure taken from Springel et al. (2005b).

and that the specific angular momentum in cold gas of disk galaxies is smaller by $\sim$40% and scatters more compared to the total dark matter halo. Steinborn et al. (2016) studied galaxies at $z = 2$ with two distinct BHs in their centre, either with one or both of them showing AGN activity, and found that BH masses were comparable if both where active. In contrast, the AGN was shown to be systematically more massive than the inactive BH in galaxies that featured both. These galaxies also showed smaller AGN accretion rates. Dolag et al. (2016) investigated the thermal Sunyaev-Zeldovich effect (Sunyaev and Zeldovich, 1970), an analysis made possible by the detailed thermal and chemical model for the intracluster medium, and compared their results to the findings of the Planck Collaboration et al. (2016). They found good agreement up to angular scales of $l \approx 1000$. Beyond that, their values overestimated the observed data significantly. Schulze et al. (2018) studied the kinematics of the simulated galaxies and found a dichotomy between

slow and fast rotators at early times, in good agreement with observations. A complete and well maintained list of all publications concerning the `Magneticum Pathfinder` simulations can be found at `http://www.magneticum.org/publications.html`.

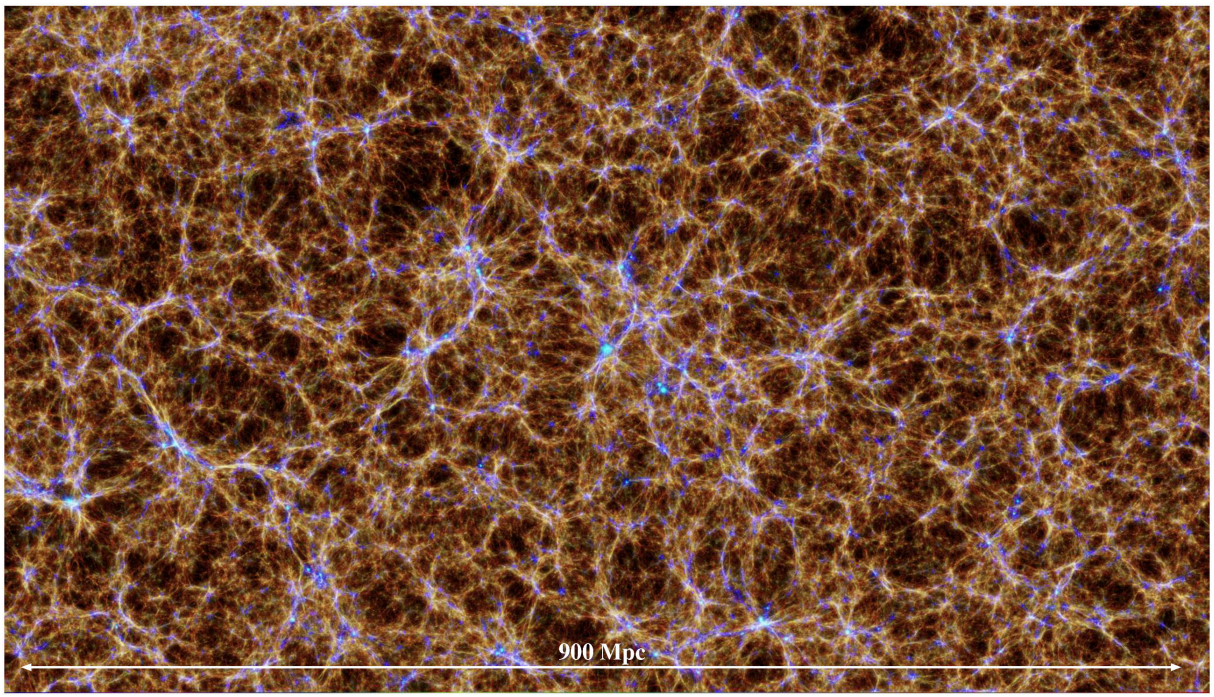


Figure 4.2: Distribution of diffuse baryonic matter at $z = 0.2$ from *Box2b/hr* of the `Magneticum Pathfinder` simulations. The colour code corresponds to the gas temperature and the visualization is centred on the most massive cluster within the box. Figure taken from Ragagnin et al. (2017).

4.8 Derived Codes

Fuelled by the solid performance of the code, both numerically and physically, a handful of spin-off codes were (and are still being) developed based upon the `GADGET` framework. Here, we present two of these.

GIZMO The `GIZMO` code (Hopkins, 2015, 2017) supplements the developer version of `GADGET-3` with several additional physics modules such as turbulent eddy diffusion models, radiation-hydrodynamics, cosmic rays, and treatment of dusty fluids (see e.g. Jalil et al. 2017 for theoretical background on the latter). More importantly, `GIZMO` offers the choice between different hydro solvers at compilation time. Users can choose between a modern SPH approach (*modern* meaning that the correction terms discussed in Sec. 2.2.5 are applied, among others), the meshless schemes MFM and MFV, as well as a fixed-grid Eulerian treatment (itself being an edge case of a meshless approach). For more details on its performance, see also Secs. 3.6 and 6.2.

The EAGLE Project Schaye et al. (2015) carried out an ambitious suite of cosmological simulations in a ΛCDM context, comparable in scale to the `Magneticum Pathfinder` simulations. While they did not make their code publicly available, they incorporated great changes to the default version of `GADGET-3`, including alternations to the time

stepping, the SPH implementation and numerous aspects of the sub-grid physics. The latter are in large parts an adaption to the methods applied in Schaye et al. (2010). See Crain et al. (2015) for an early study concerning the parameter space of `EAGLE` and `http://icc.dur.ac.uk/Eagle/publications.php` for a list of publications using their simulated data.

5 Implementing an MFM Scheme in GADGET

Our meshless scheme of choice for implementation in `OpenGadget3` is MFM rather than MFV for its convenient feature of preserving masses, facilitating coupling to gravity, as well as for the slightly better performance that Hopkins (2015) found upon thorough direct comparison.

In this section, we will first describe the most important facets of our realization of the MFM scheme in `OpenGadget3` and afterwards present various numerical tests that we carried out, each highlighting different aspects of the implementation.

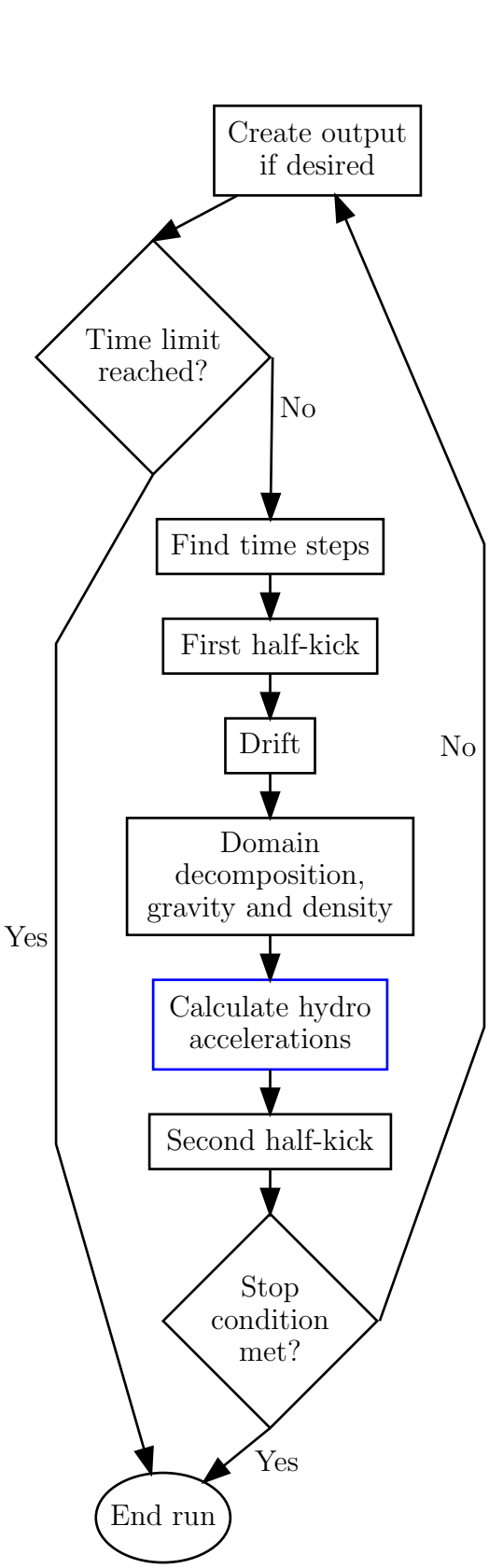
5.1 Code Structure

The MFM package in `OpenGadget3` is written in the `C++` programming language (as opposed to `C` which is used for the majority of other functionalities). The first reason for this decision is based on the additional capabilities of `C++` over its predecessor, most notably the object oriented approach of the language. For example, we realized different Riemann solvers as child classes of a base class `RiemannSolver`, facilitating the implementation of new solvers a great deal. The second reason is of purely practical nature, namely to allow for an easy porting of the MFM scheme applied in `GANDALF` (Hubber et al., 2018), a code natively written in `C++`, which we used as a basis for our implementation. Luckily, this decision does not disrupt workflows, since `OpenGadget3` ought to be compiled with `C++`-compatible compilers in any case to make use of their numerous advancements.

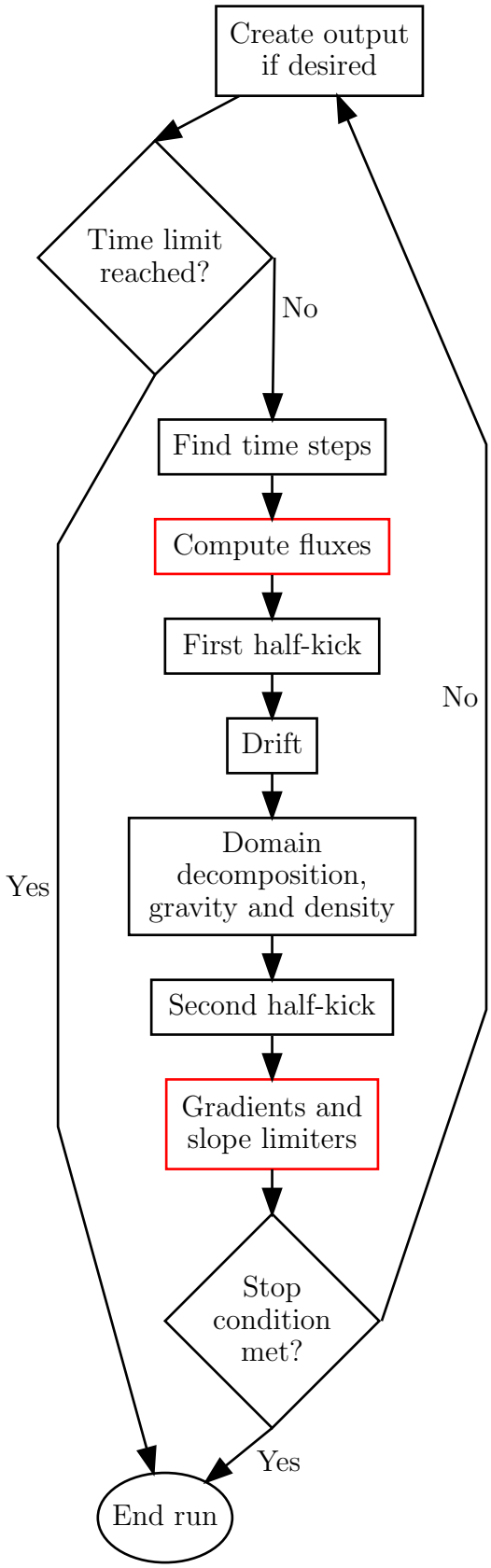
To enable `OpenGadget3` to use MFM instead of SPH, a switch can be set in the configurations file, resulting in the differentiation between solvers at compile time. Additional switches may be set to choose the slope limiter and Riemann solver to be applied. The parameter file, specifying the characteristics of each run, such as the initial conditions (ICs) file, the output frequency, and the total runtime, may be left untouched (other than adjusting the neighbour number to the kernel used). Parameters pertaining to SPH specific parts are simply ignored.

Fig. 5.1 illustrates how `OpenGadgets`'s main loop (iterating over all time steps) is altered when MFM is chosen over SPH. After each particle was assigned a time step (see Sec. 4.3), the fluxes between neighbouring particles are calculated if at least one of them is currently active. We allow for parallel computation using the message passing interface (MPI) framework by creating appropriate, reduced data structures for every gas particle that can easily be exchanged between CPUs. Since conservation demands $\vec{F}_{ij} = -\vec{F}_{ji}$, we can reduce the number of iterations necessary to calculate all fluxes by almost half. We do so by skipping each neighbour j of a target particle i if $(\Delta t)_j < (\Delta t)_i$, knowing that this flux will ultimately be calculated when j becomes the target particle. The resulting fluxes are always recorded for the target particle and, if $(\Delta t)_j > (\Delta t)_i$, also for the neighbour particle (with opposite sign). Only if both particles are in the same time bin will the two fluxes be calculated independently. We accept this penalty in computation time to avoid the necessity of bookkeeping on which neighbour pairs on equal time steps have already been evaluated. All calculated fluxes are multiplied by the current time step of the target particle and accumulated in a $\mathrm{d}\vec{Q}$ variable. During the ensuing first half-kick, these variables are adjusted for gravitational contributions for all active particles.

The following drift routine remains untouched by our implementation, as do the domain decomposition and the gravity update. Likewise, the density calculation using



(a) Main loop with SPH



(b) Main loop with MFM

Figure 5.1: Comparison of `OpenGadget3`'s main loop using either SPH (a) or MFM (b) as a hydro solver, unique routines are highlighted. These massively simplified diagrams ignore any physics other than gravity and fluid interactions.

the SPH framework is performed for all active particles and the primitive vectors $\vec{W}$ are updated accordingly.

The second half-kick resembles the first one in that we update the $\mathrm{d}\vec{Q}$ by gravitational accelerations. At this point, we also carry out the update of the $\vec{Q}$ vectors of active particles as $\vec{Q}_{\mathrm{new}} = \vec{Q}_{\mathrm{old}} + \mathrm{d}\vec{Q}$. Afterwards, all affected $\mathrm{d}\vec{Q}$ are zeroed again, the primitive vectors are recalculated from the updated conservative vectors, and the entropy is trivially calculated from the internal energy.

This is followed by the computation of gradients. Similar to the flux computation, we make full use of the capabilities of MPI to allow for parallel execution. We apply the formalism introduced in Sec. 3.2.2 and evaluate it for all active particles. As a safeguard, we calculate the squared condition number $N^2_{\mathrm{cond},i}$ as

$$N^2_{\mathrm{cond},i} = \frac{\left\|\underline{\underline{E_i}}\right\| \cdot \left\|\underline{\underline{B_i}}\right\|}{d^2} \tag{5.1}$$

with dimensionality d as well as

$$\left\|\underline{\underline{E_i}}\right\| \equiv \sum_{\alpha=1}^{d}\sum_{\beta=1}^{d}\left|\underline{\underline{E_i}}^{\alpha\beta}\right|^2 \tag{5.2}$$

and accordingly for $\|\underline{\underline{B_i}}\|$. $N^2_{\mathrm{cond},i}$ is a measure for how much errors propagate during matrix inversion. In practice, a large condition number is usually the result of an ill-defined gradient in one spatial dimension due to a lack of neighbouring particles in this direction. If we identify a value of $N^2_{\mathrm{cond},i}$ above a certain threshold, we do not use the gradients obtained in this manner but rather fall back on using the less accurate but more stable gradients obtained from the SPH-like density calculation. At this point, we also check if any particles ought to be *woken up*, see Sec. 5.4 for more details. Lastly, the calculated gradients are flattened (if necessary) with the chosen slope limiting scheme, once again exploiting MPI.

Afterwards, the code checks if any special stop conditions are met (such as a manual abort or a depletion of assigned CPU time) and enters the loop anew if that is not the case, advancing the physical time t by the time step Δt of the previous iteration.

5.2 Treatment of Discontinuities

First, we examine how our meshless scheme fares in dealing with discontinuities by performing Sod shock tube tests. Studied extensively by Sod (1978), these numerical problems are characterized by ICs that are split in two distinct regimes: a left state $\vec{W}_L$ and a right state $\vec{W}_R$. By convention, we set $\rho_L = 1$ and $\rho_R = 0.125$ as well as $\vec{v}_L = \vec{v}_R = 0$. We also fix the pressure of the left state p_L ad hoc and apply a double root-finding algorithm to calculate p_R such that a target Mach number of the shock M will be reached. All physics other than hydrodynamics are switched off. The problem can be solved in one, two, or three dimensions[17]; we choose the latter for a more realistic test. In each case, the shock will be aligned along the x-axis. Since we use periodic boundary conditions,

[17]Note that Riemann solvers (Sec. 3.3) are themselves nothing else than algorithms solving the shock tube problem in 1D. When assessing how our MFM implementation captures these in 3D we are thus, in a sense, testing how well these particle-to-particle solutions scale to more dimensions and bigger scopes.

we are in actuality dealing with two discontinuities. To avoid self-interactions we extend the computational domain generously in x-direction, ultimately obtaining a box of side lengths $[140, 1, 1]$ with the left and right initial state separated at $x = 70$. Both sides of this tube are constructed by stacking cubes of 7^3 particles that have previously been relaxed with the chosen kernel. One the right side, we simply repeat the cube 70 times. To maintain constant particles masses, eight of these cubes comprise a single stacked cube on the left side (resulting in an eightfold increase in density) which is then repeated 70 times as well. We let the system evolve and compare our results to those obtained with the AMR code `Athena++` (White et al. 2016, Felker and Stone 2018) and to the analytic solution (which we calculate with the approach outlined in Pfrommer et al. 2006).

We start by choosing $p_L = 200/3$ and $M = 1.5$, resulting in $p_R \approx 8.154$. This scenario represents mild shocks that we expect to see plentiful in cosmological simulations. We choose $C_{\mathrm{CFL}} = 0.05$, a fairly low value which, however, allows us to reduce the noise in MFM. We use the exact Riemann solver by Toro (1999), a slope limiting scheme following Springel (2010), as well as the C^6 kernel with 295 neighbours. Fig. 5.2 shows the outcome of these tests. The three regions of interest will be referred to as the rarefaction wave (from $x \approx 54$ to $x \approx 67$), the contact discontinuity ($x \approx 80$), and the shock discontinuity ($x \approx 93$).

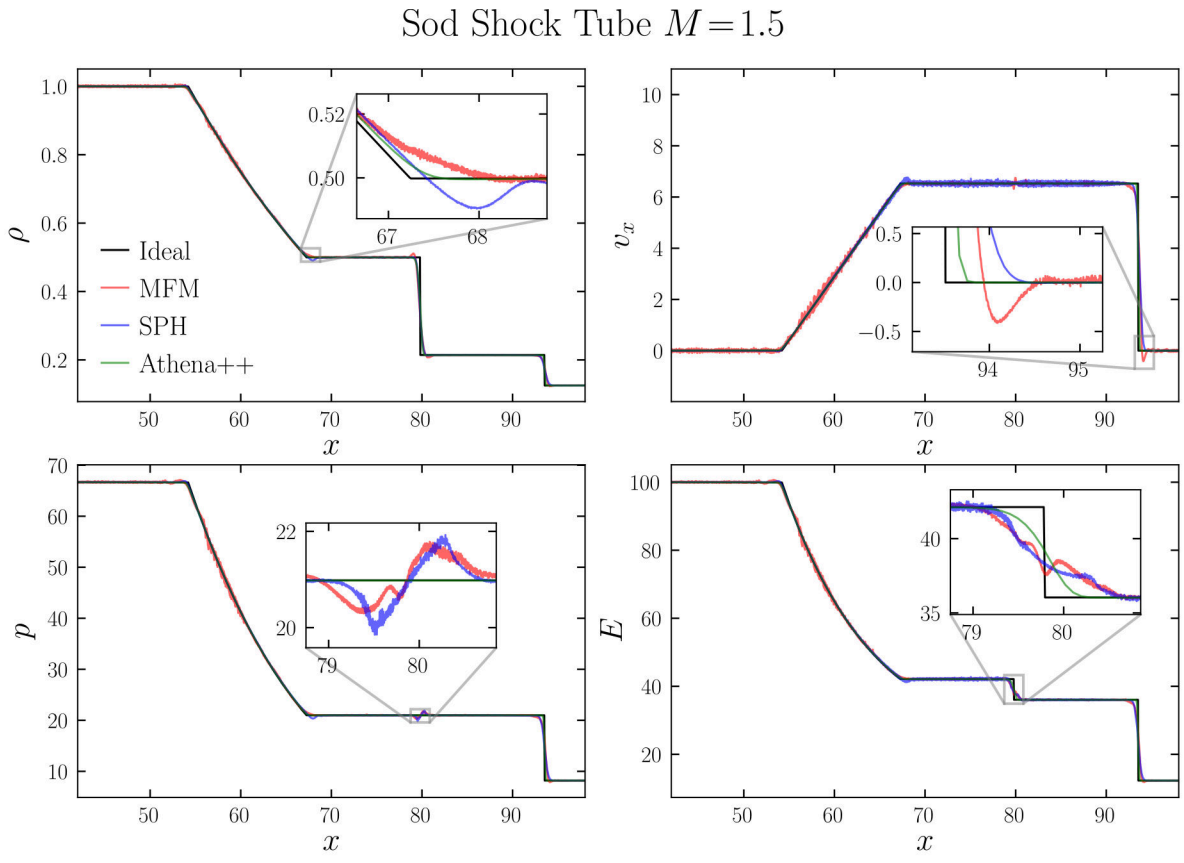


Figure 5.2: Sod shock tube with a Mach number $M = 1.5$ at $t = 1.5$. The SPH and MFM runs in `OpenGadget3` were executed with the same parameters and ICs. The ideal solution is calculated according to Pfrommer et al. (2006). The solution of `Athena++` was achieved using $800 \times 1 \times 1$ cells and with a considerably higher C_{CFL} (by a factor of eight).

Overall, we see that SPH and MFM yield similar results although a thorough analysis reveals differences in certain details. For example, the beginning of the rarefaction wave

(see detail in the $\rho(x)$ panel, upper left) is better resolved in MFM and does not feature the undershoot of the SPH solution. However, MFM shows considerably more noise despite the low C_{CFL} we chose. This is a consequence of the lack of artificial viscosity, leading to smoother solutions in SPH. The detail in the $p(x)$ panel (lower left) shows the *pressure blip*, an artefact of the inconsistency between the smoothed-over discontinuity in ρ and the sharp discontinuity in p in the ICs. While it is present in both solutions, its amplitude is slightly reduced in MFM. The pressure blip also leads to an over-smoothed contact discontinuity (detail in $E(x)$ panel, lower right) present in both schemes. Lastly, the shock discontinuity shows a slight undershoot of the MFM solution, most notably in the velocity (detail in $v_x(x)$, upper right panel).

We also simulate more violent shocks, setting once again $p_L = 200/3$ but this time choosing $M = 100$, yielding $p_R \approx 0.001$. All remaining parameters are kept constant. While Mach numbers of this scale are generally much rarer, they are expected in accretion shocks of galaxy clusters (see e.g. Skillman et al. 2008 and Vazza et al. 2015a,b). Fig. 5.3 shows the outcome of this simulation.

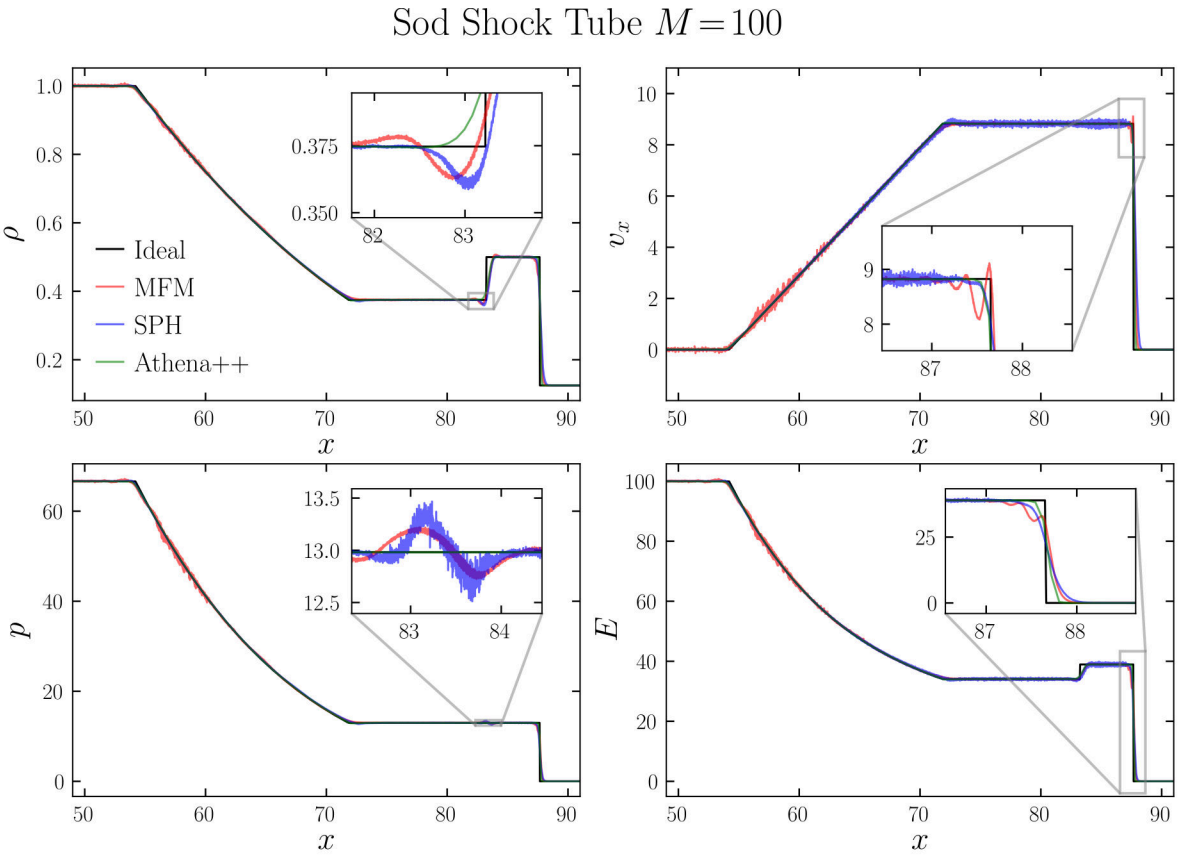


Figure 5.3: Sod shock tube with a Mach number $M = 100$ at $t = 1.5$. Apart from a different p_{R} in the ICs, the same parameters as in the previous run (Fig. 5.2) were used.

While the density is, in general, still more noisy in the MFM scheme, we see that the pressure within the shock region is better resolved with our meshless approach, leading to a pressure blip that is both smoother and less pronounced than in SPH. This, in turn, leads to a slightly better approximation of the contact discontinuity (detail in $\rho(x)$) even though both approaches do produce an undershoot. The contact discontinuity, however, still shows spurious features in MFM, this time more clearly visible in the downstream

region (details of $v_x(x)$ and $E(x)$). We also observe a region of increased noise within the rarefaction wave both in this and the previous simulation.

We repeated all tests with the exact same parameters but using the M^4 kernel with 32 neighbours instead. The results were qualitatively similar to those shown in Figs. 5.2 and 5.3 but featured considerably more noise both with SPH and MFM. The next section will illuminate the choice of kernel functions in more detail.

We conclude that our MFM implementation performs about as well as SPH when using equal parameters. The biggest drawback we found was the considerable increase in noise, especially in the density estimates. This is also observable in regions not yet penetrated by the shock. Lowering C_{CFL} allows us to lessen this effect but for any choice, SPH still delivers better (i.e. smoother) results with the C^6 kernel. However, our MFM approach resolves discontinuities slightly better and shows less over-smoothing. An exception from this is the shock discontinuity that is resolved worse in MFM. We expect this issue to be addressable by a better suited choice of slope limiting procedure but did not explore this avenue any further.

We also note that the solution of `Athena++` performs much better than both schemes while taking mere seconds to compute. This, however, is simply attributed to the one-dimensional nature of these runs[18], combined with a much larger C_{CFL}. In addition, the Riemann solver is only called between direct neighbour cells whereas in `OpenGadget3` it is invoked for each active particle (roughly) N_{Ngb} times. Hence, the solution obtained with the AMR code resembles the solution of a single Riemann solver call much more closely.

5.3 Kernel Functions and Neighbour Numbers

We test the hypothesis that MFM would allow us to use both a simpler[19] kernel function as well as a smaller number of neighbours.

To this extent, we set up a Sedov blast wave test after Sedov (1959): Inside a region of homogeneous density (which we realize through a perfect grid of particles) we insert an amount of thermal energy. The energy may either be associated to one particle alone or spread out over a kernel — we choose the latter. This setup leads to a blast wave expanding from the point of injection outwards. We compare the results of the simulation to the analytical solution by Book (1994).

Fig. 5.4 shows the outcome of our tests. Depicted are radial profiles of a 3D cube containing 64^3 particles, simulated both with SPH and MFM as well as with the C^6 kernel (with 295 neighbours) and the M^4 kernel (32 neighbours)[20] each. The first striking observation is that the C^6 kernel with its significantly larger neighbour number (by almost a factor of ten) shows considerably less scatter than the M^4. Comparing the performance of the C^6 kernel in the two schemes shows that they yield comparable results with the most noteworthy difference that MFM slightly decreases the width of the smoothed wave preceding the actual shock front (see the regime at $r \approx 0.4$). Changing the kernel drastically

[18] Due to the Eulerian nature of the code, the solution for this particular problem would virtually be the same in two and three dimensions.

[19] We quantify the *simpleness* of a kernel by the inverse of its highest order polynomial. The higher this value, the easier (i.e. quicker) the computation of each $W(\Delta\vec{r}, h)$ becomes.

[20] As pointed out by Dehnen and Aly (2012), the M^4 kernel becomes susceptible to pairing instabilities for $N_{\mathrm{Ngb}} > 55$ whereas the C^6 heavily benefits from an increased neighbour number. Our exact values are taken such that the full width at half maximum, given $h(N_{\mathrm{Ngb}})$, equals that of a pure Gaussian in a field of constant density.

increases the noise in the SPH approach. While MFM also sees an increase in scatter, it is much less significant. As can be seen most prominently in the density profile, the run using MFM and the M^4 kernel approximates the analytic solution of the shock best while further reducing the effect of the leading smoothing wave.

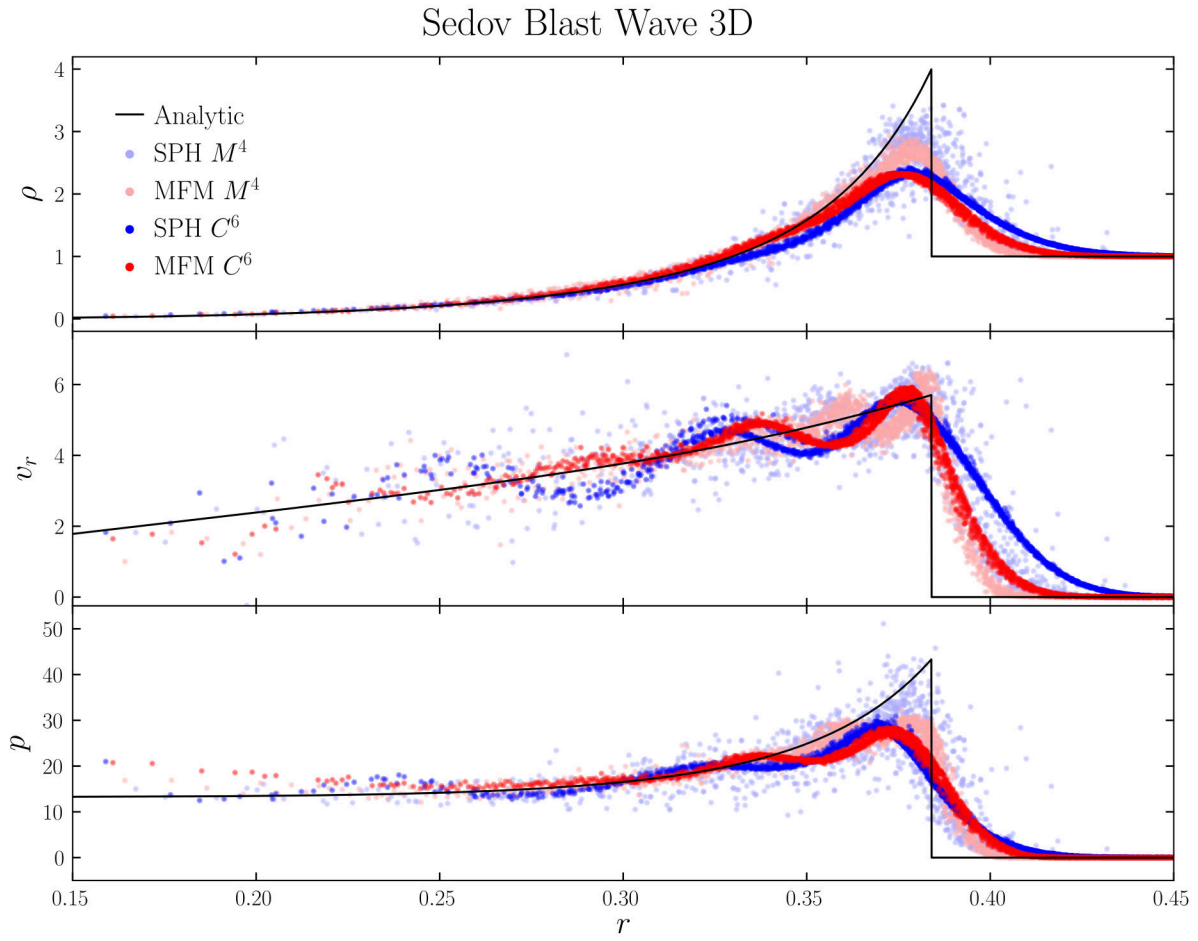


Figure 5.4: Sedov blast wave in 3D, simulated from an initial regular grid of 64^3 particles with internal energy injected (in a kernel-weighted manner) in the most central 32 particles. We plotted every 50^{th} particle after sorting them according to their distance r to the centre of the initial energy injection. v_r in the second panel is the radial velocity with respect to this point. The M^4 kernel (Eq. 2.23) used 32 neighbours whereas the C^6 kernel (Eq. 2.25) used 295. For the MFM runs, we applied the exact Riemann solver by Toro (1999) and followed the slope limiting procedure of Springel (2010).

The phenomenon of a larger number of neighbours leading to less noise is very much expected as greater neighbour numbers lead to increased smoothing lengths and hence smoothing over a larger radius, effectively equalizing physical properties on larger scales. We also expect to see less over-smoothing with MFM since meshless schemes reduce the prevalence of the kernel to the domain partitioning alone[21] (see Sec. 3.2.1), thereby alleviating the effect of the smoothing procedure. The reason for the superior performance of the C^4 kernel in MFM, however, may not be so obvious.

[21]This is not exactly true in our implementation since we still use a density loop performed in an SPH-like manner. This lessens, but does not invalidate, the argument.

Fig. 5.5 illustrates why, in contrast to SPH, a higher (i.e. too high) number of neighbours leads to declining results using MFM. Nevertheless, this does not imply that we can just reduce the number of neighbours to an arbitrary degree as Fig. 5.6 shows.

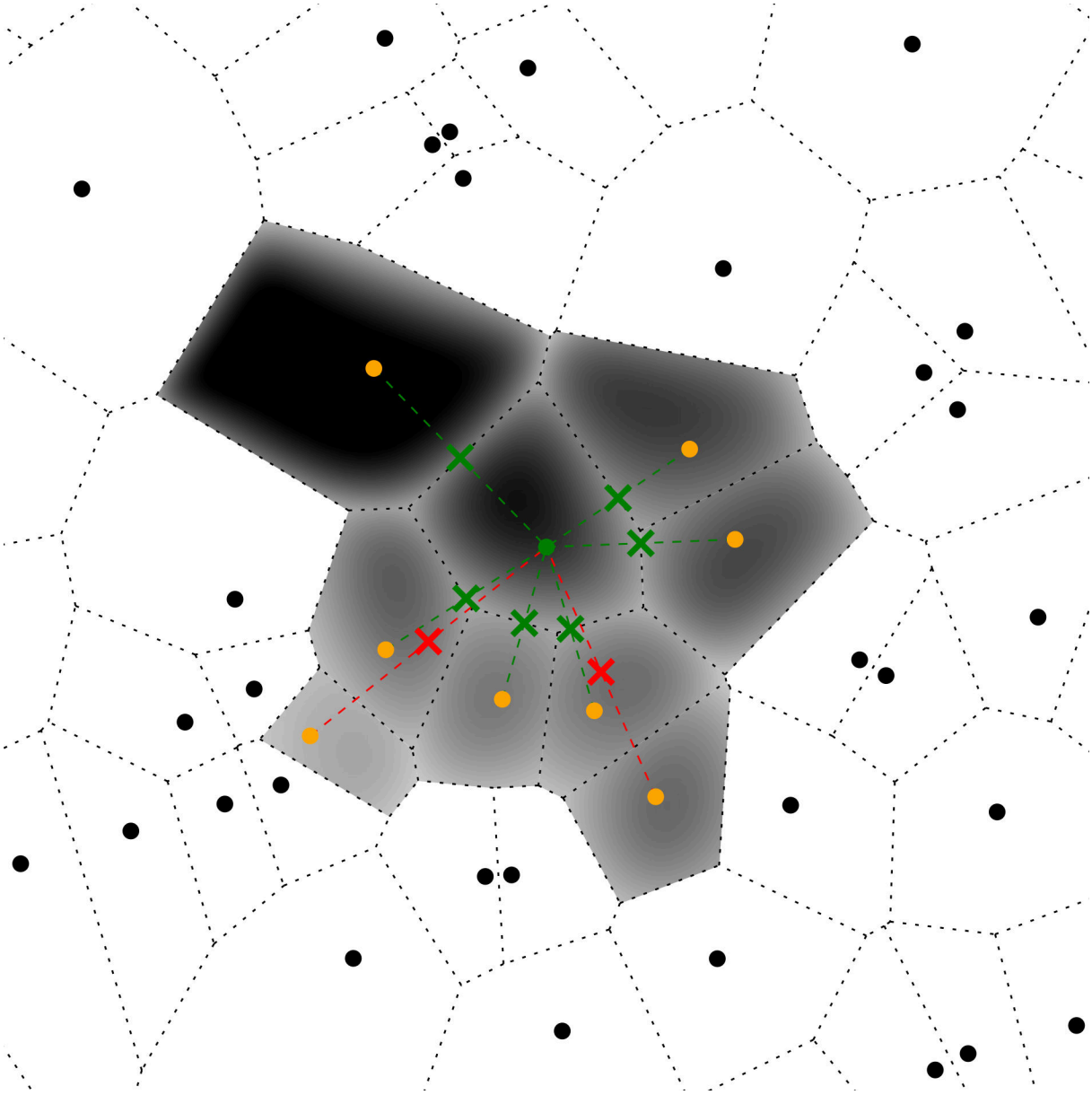


Figure 5.5: Repetition of Fig. 3.2, this time taking into account more neighbouring particles (based on proximity to the target particle). As indicated by the red crosses, the fluxes with particles that do not share a common boundary with the target particle are evaluated within the domain of other particles, tarnishing their accuracy.

In summary, we find that a simpler kernel with a lower neighbour number indeed leads to better results. This, however, comes with the negative side effect of increasing the noise of MFM even further. To offset this, a lower C_{CFL} may be chosen.

5.4 Wake Up Scheme

Time stepping in our MFM scheme follows the principles that we outlined in Sec. 3.5. In a purely hydrodynamical simulation, the only time step criterion we use is given by Eq.

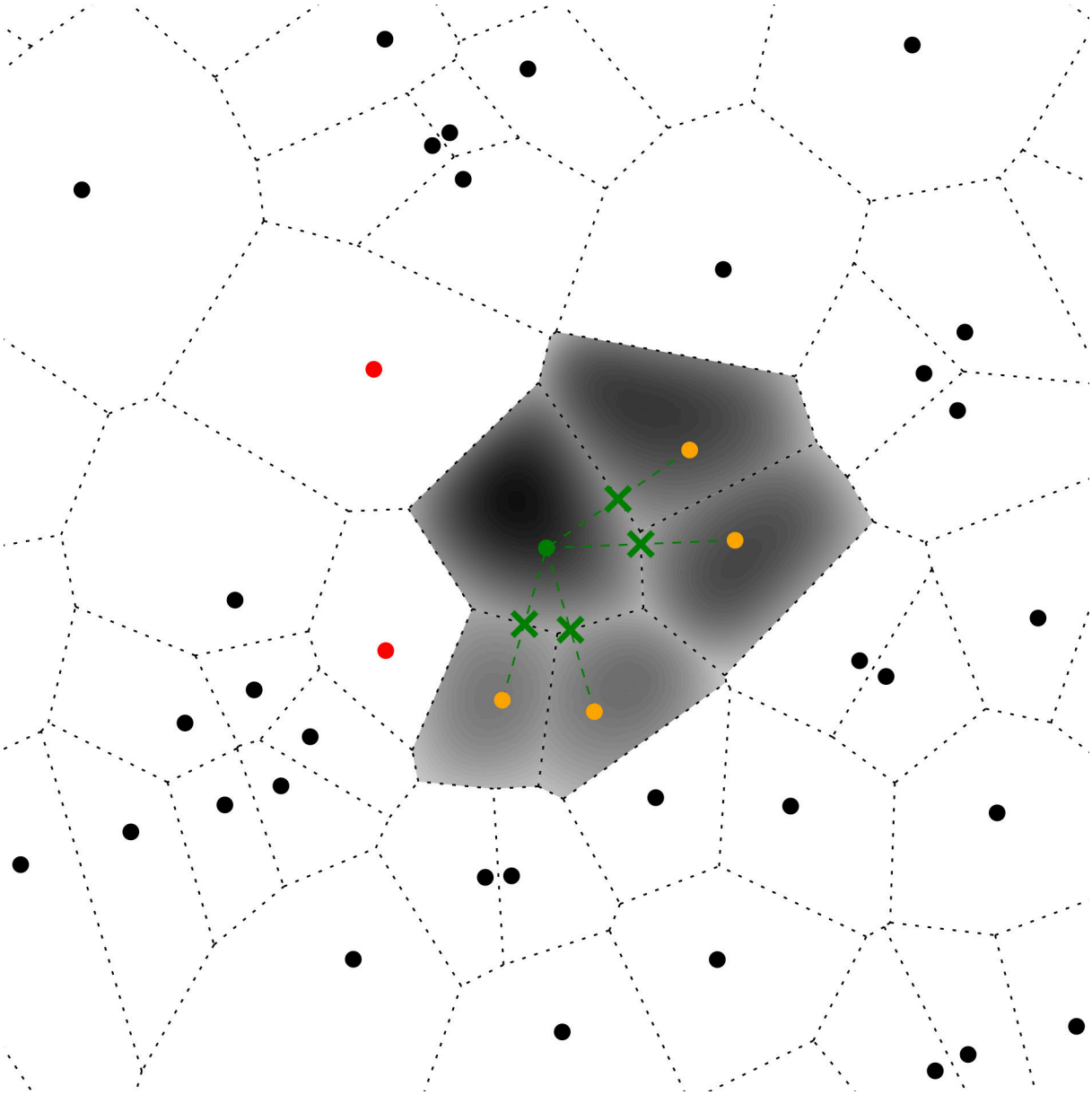


Figure 5.6: Repetition of Fig. 3.2, this time taking into account fewer neighbouring particles. The two particles in red are now not counted as neighbours and fluxes between them and the green target particle are neglected, introducing random errors. If the fluid features a density gradient, a common occurrence in cosmological simulations, this would lead to a systematic directional bias instead.

(3.25). The resulting $(\Delta t)_i$ are then converted to a time bin in a power-of-two time step hierarchy and the fluxes are solved accordingly.

While the above approach works reasonably well in most cases, it does struggle to accurately reproduce shocks. Per definition, these feature particle velocities in excess of their sound speed, leading to a propagation of information not taken into account in our simple approach. This in turn may lead to particles on largely different time steps coming close to each other and, ultimately, particles on small time steps penetrating a regime of particles on larger time steps without the latter accurately reacting to their presence until much later (i.e. when they become active themselves).

To prevent these fringe cases from happening, we applied a wake up scheme similar to the one of the SPH mode of `OpenGadget3`: Each particle is assigned a `wakeup` attribute, a flag indicating whether the particle should enter the wake up routine which is called during the determination of time steps at the beginning of each iteration of the main loop (cf. Fig. 5.1). Initially, these flags are unset for all particles. During the gradient computation[22] (the last *physics* routine in the main loop) we perform the check

$$v_{\mathrm{sig},ij} \overset{?}{>} \mathcal{W}\, v_{\mathrm{sig},j} \tag{5.3}$$

for each pair of target particle i and neighbour particle j with

$$v_{\mathrm{sig},ij} = c_{\mathrm{s},i} + c_{\mathrm{s},j} - \min\left[0, \frac{\left(\vec{v}_i - \vec{v}_j\right)\cdot\left(\vec{r}_i - \vec{r}_j\right)}{\left|\vec{r}_i - \vec{r}_j\right|}\right] \tag{5.4}$$

the signal velocity between i and j (the last addend taking into account their relative movement), $v_{\mathrm{sig},j}$ the maximum of which detected for particle j (cf. Eq. 3.26), and $\mathcal{W} \geq 1$ a parameter. Note that while particle i must be active to enter the gradient computation in the first place, this is not necessarily true for particle j. Hence, the maximum signal velocity of j stems from the last time step when this particle was evaluated. If the condition in Eq. (5.3) is met we know that the signal velocity in the vicinity of j has increased since it was last estimated and the particle ought to be woken up. $\mathcal{W}$ can be used to control the aggressiveness of the wake up scheme (with higher values corresponding to a higher wake up threshold) and is set at compilation time.

We test the effects of the wake up scheme by once again simulating a Sedov blast wave with MFM. This time, the simulations are performed in 2D with 64^2 particles initially positioned along a regular grid. We choose a very conservative Courant-Friedrichs-Lewy number of $C_{\mathrm{CFL}} = 0.025$ to exclude errors caused by too small temporal resolution. We perform the simulation with and without using the wake up scheme. In the former case, we choose a wake up parameter of $\mathcal{W} = 3.0$.

Fig. 5.7 shows the evolution of the blast wave in physical space; the particles are colour coded according to their current time step with larger $(\Delta t)_i$ corresponding to lighter colours. Already at very early times (leftmost panels) we see that the wake up scheme adjusts time steps in a large radius around the blast. In contrast, the simulation without wake up only evaluates particles very close to those that where initially excited more often. As the simulation progresses, the spurious effects of this become clear: At $t = 0.004$ (central panels), when ignoring wake ups, the energetic particles penetrate the surrounding regular grid without the grid reacting to their presence. As the rightmost panels show, this ultimately even breaks the radial symmetry of the solution.

Fig. 5.8 furthers this observation: Early on during the simulation, the run without wake ups transforms almost half of the total energy budget $E_{\mathrm{total},0}$ into kinetic energy E_{kin} — the central particles are accelerated but feel little to no pushback from their surroundings. The turnaround of this trend coincides exactly with the maximum permissible time step we set for this test. Only at this time, the time steps of all particles are re-evaluated. This behaviour is mirrored two more times, albeit less pronounced. Eventually, both runs

[22] We could, in theory, perform a direct check for too large time step differences after the time step assignment instead. This, however, would require an additional double loop over all particles and their respective neighbours. Since such a loop is necessary when calculating gradients in any case, we save computation time by checking for the wake up condition there.

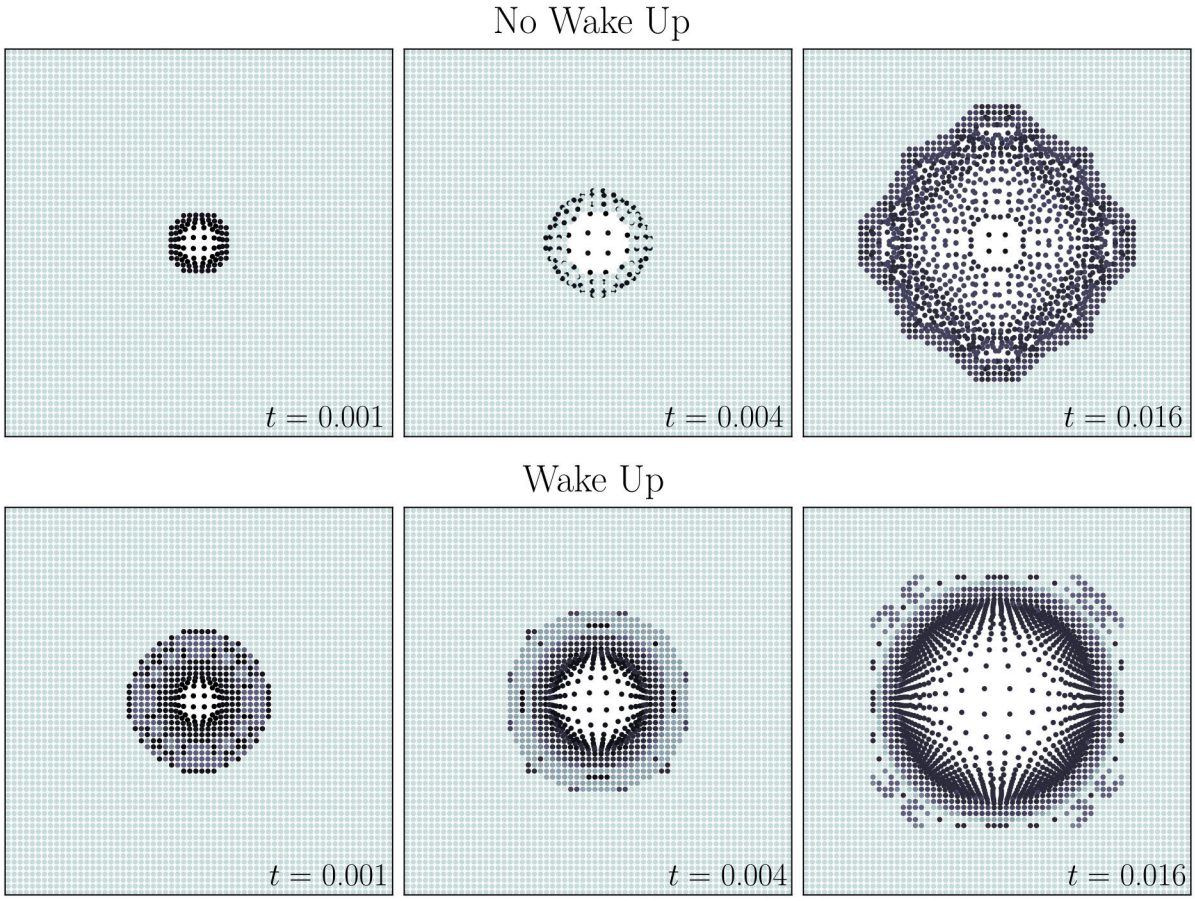


Figure 5.7: Evolution of a Sedov blast wave in 2D, initialized as a regular grid of 64^2 particles. Shown are three snapshots at distinct times, both when implementing the wake up scheme ($\mathcal{W} = 3.0$, lower panels) and without it (upper panels). The colour code of the particles corresponds to their time step — darker colour indicate smaller values of $(\Delta t)_i$. To ensure numerical stability also in the case of no wake up, a neighbour number of 24 was used, a value considerably larger than the ideal value for the M^4 kernel in 2D ($\sim$12).

approach a similar ratio of kinetic energy to thermal energy $E_{\rm th}$. The small deviations from the total energy (which should be conserved) at the beginning of the simulation (see detail) is attributed to the asynchronous updating of active and inactive particles.

We conclude that the implementation of the wake up scheme significantly improves the outcome of the simulation. Since the maximum signal velocities $v_{{\rm sig},i}$ are computed in the gradient routine in any case, the additional steps required are miniscule. We note, however, that our implementation of the wake up scheme introduces spurious fluxes. As described in Sec. 5.1, we accumulate flux contributions in $\mathrm{d}\vec{Q}$ values and only update the $\vec{Q}$ vector once a particle is active. During each flux evaluation, these $\mathrm{d}\vec{Q}$ are increased by $\vec{F}_{ij}\cdot(\Delta t)_i$ (i being the active particle). While $\vec{Q}_i$ is updated during the same time step, the contributions to an inactive particle j are stored by summing them up into a single value of $(\mathrm{d}\vec{Q})_j$. If the time step of the (previously) active particle i is now reduced even further due to it being woken up *before* particle j becomes active again, the contribution of i to $(\mathrm{d}\vec{Q})_j$ will be counted again (albeit only a fraction of it). Despite our best efforts, we did not find a compelling solution to this problem. The biggest hurdle in setting up a

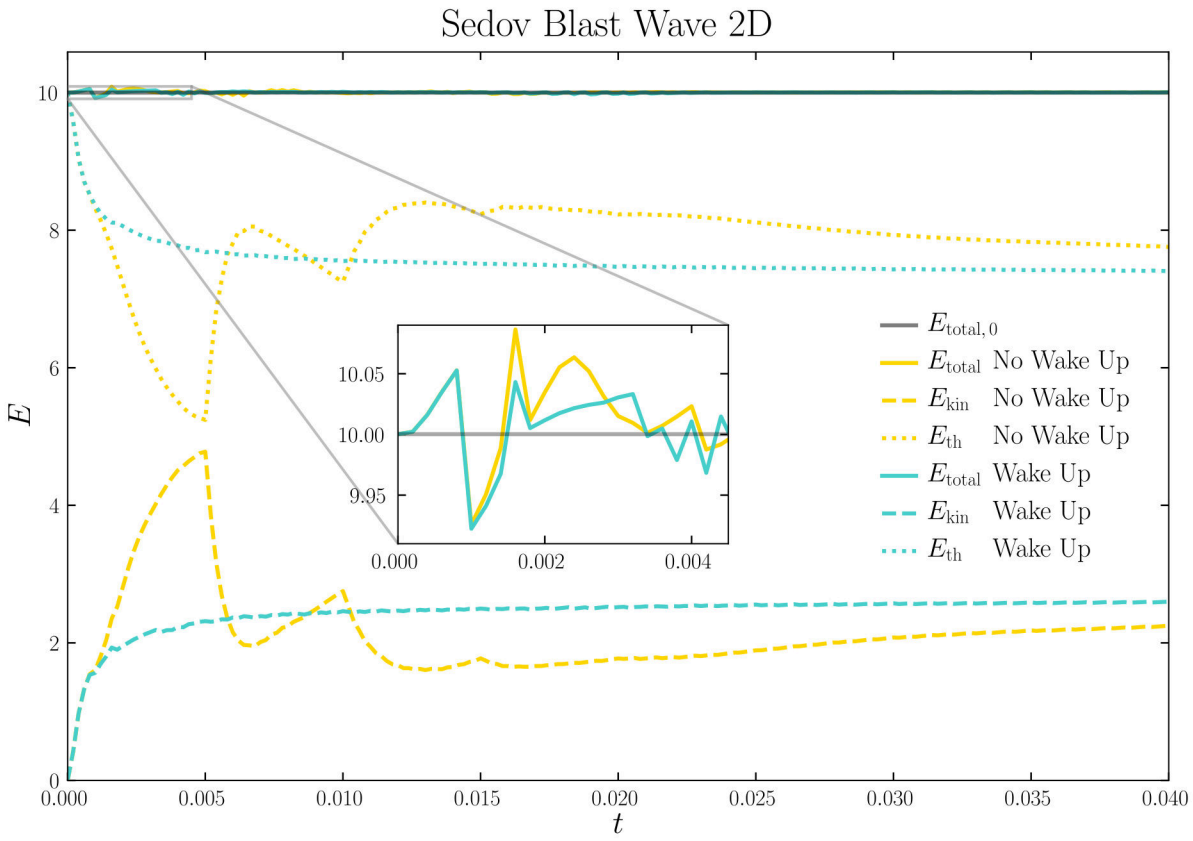


Figure 5.8: Evolution of the energy components of the simulations depicted in Fig. 5.7.

correction is the lack of stored previous data. Since a $\mathrm{d}\vec{Q}$ vector cannot be divided into its contributions retroactively, we cannot simply subtract the superfluous contributions, and since old $\vec{W}$ or $\vec{Q}$ vectors are not stored either[23], we are unable to recalculate them. Nevertheless, we did not find a significant impact of these errors on our simulations and it is debatable if the improvements in accuracy would justify the considerable additional storage space required.

All simulations in upcoming sections, as well as those already shown in Secs. 5.2 and 5.3, made use of the wake up scheme.

5.5 Fluid Mixing

We test the efficiency of fluid mixing by studying how well our implementation can resolve Kelvin-Helmholtz instabilities (see e.g. Tian and Chen 2016). To this extend, we set up ICs in a sheet with dimensions $[256, 256, 8]$. We separate our particles in two regimes A and B with transitions between the two at $y = 64$ and $y = 192$ (periodic boundary conditions enabled). The density of particles in B equals roughly double the density of particles in A ($\rho_{\mathrm{B}} \approx 2\rho_{\mathrm{A}}$) whereas the pressure is constant over both phases. Additionally, all particles in A are assigned an x-velocity $v_{x,\mathrm{A}} > 0$ and all particles in B the same, but opposite velocity ($v_{x,\mathrm{B}} = -v_{x,\mathrm{A}}$). For all particles $v_y = v_z = 0$. With this setup, vortices where both phases are intertwined should form at the transition layers.

[23] A rigorous solution would also require to store the times at which each $\vec{F}_{ij}$ was last calculated, as well as the positions of particles i and j at that time.

The outcome of these tests is depicted in Fig. 5.9. As can be seen clearly, in a pure SPH approach the two phases remain well separated. Although the contact layer is deformed, the spurious pressure terms in SPH suppress any mixing effects (see Sec. 3.1.2). Invoking the correction terms for SPH introduced by Beck et al. (2016) (most notably the artificial viscosity and conduction) changes the outcome: The scheme becomes more dissipative and a transition layer between the phases A and B can be observed. Applying our MFM implementation reinforces this behaviour quite significantly. We prominently observe a fuzzy and broad transition between both phases. It should be noted that the outcome is more noisy, even in regimes of constant density. We attribute this to the fact that we used the M^4 kernel with 32 neighbours (as opposed to the C^6 kernel and 295 neighbours in SPH), naturally leading to less smoothing and more small-scale variations.

We stress that performing these tests highlighted the importance of checking the condition number after the gradient estimation, see Eq. (5.1). Amplified by the regularity of the grid, the ensuing shear flows at the transition layers sporadically led to ill-defined gradients in perpendicular directions and exceeded the threshold for N^2_{cond}. In such cases, we disregarded the gradients calculated with MFM and used those generated during the (SPH-like) density loop instead. These are less accurate and lead to more noise — however, this fallback scheme was invoked seldom enough that we do not expect any significant deterioration in accuracy.

5.6 Runtime

In addition to the qualitative performance of MFM, we also study how much computation time our scheme uses compared to SPH, making use of already simulated test suites. We also study the impact of exchanging the applied Riemann solver.

Tab. 5.1 shows how long each combination of Mach number M, kernel, and hydro scheme took to simulate a Sod shock tube as described in Sec. 5.2. Unsurprisingly, the Mach number does not impact the runtime significantly. With the M^4 kernel and constant C_{CFL} and N_{Ngb}, MFM runs roughly 20-25% slower than SPH. When changing to the C^6 kernel and increasing the number of neighbours by a factor of ~ 9, the slowdown increases to about 40%, indicating a worse scaling with neighbour number for MFM than for SPH. This and the overall faster performance of SPH over MFM with similar parameters is attributed to the necessity of repeatedly calling a Riemann solver in the latter, a procedure more time consuming than the calculation of hydrodynamical forces in SPH. Additionally, this force calculation requires a single loop over all active particles and their neighbours whereas in MFM the flux computation, the gradient estimates, and the slope limiters each require such a loop (cf. Fig. 5.1).

As we showed in Sec. 5.3, however, one advantage of MFM over SPH is that it allows for less neighbours to be used while still achieving good (if not better) results. A more appropriate comparison would thus be to contrast the performance of SPH using the C^6 kernel with that of MFM and the M^4 kernel, yielding a speed-up by a factor of ~ 2 when using the meshless approach. We note, however, that all our tests showed that such a comparison leads to considerable more particle noise in the solution obtained with MFM. To offset this, we found a decrease of C_{CFL} to be most effective. This has the added advantage that C_{CFL} scales rather directly with the total computation time of the code, allowing for a precise fine-tuning of accuracy to the allotted CPU time for production runs.

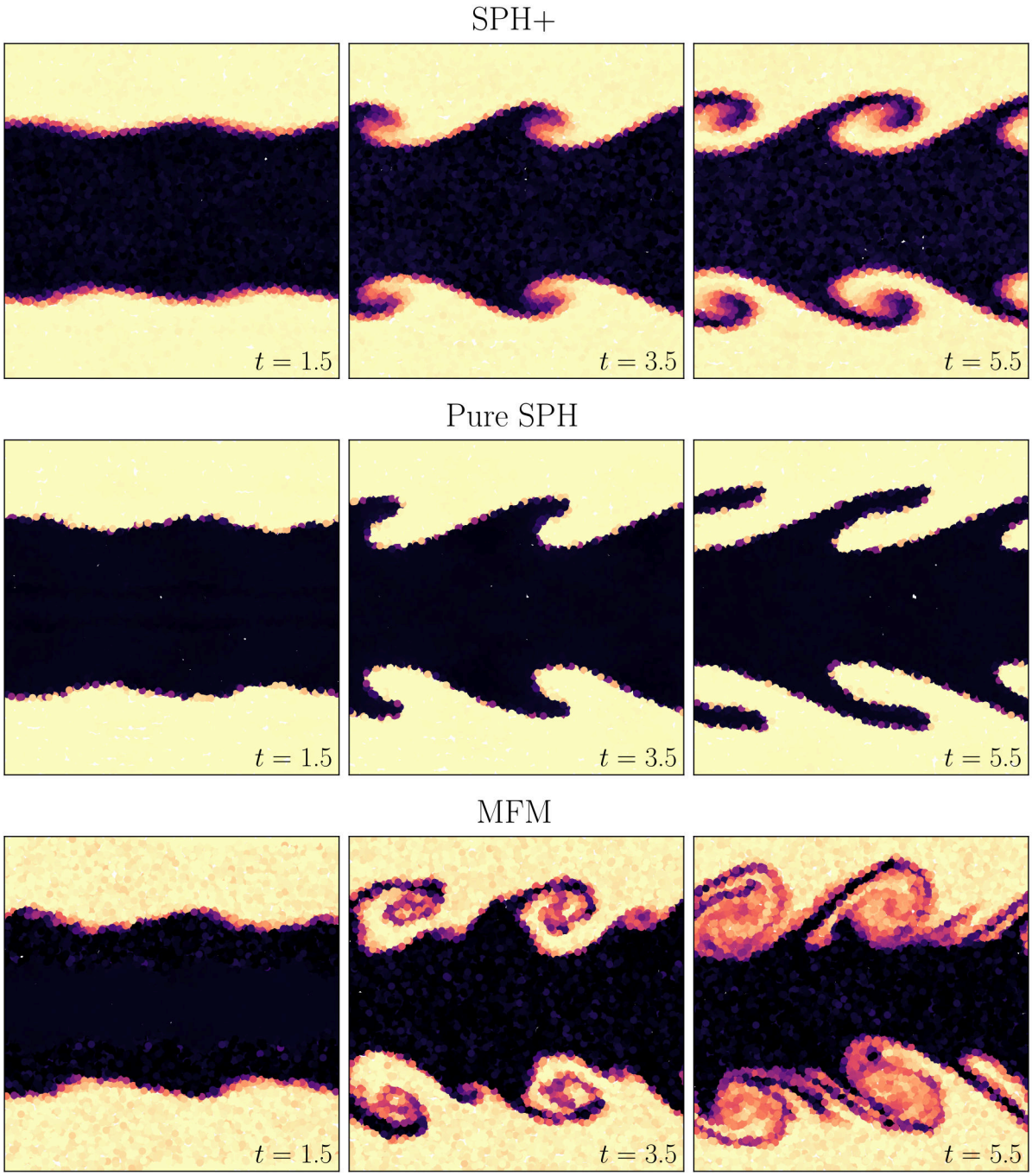


Figure 5.9: The Kelvin-Helmholtz instability as simulated using SPH with correction schemes described in Beck et al. (2016) (*SPH+*, upper panels), SPH without any corrective terms (middle panels), and with MFM (lower panels). Both SPH runs used the C^6 kernel with 295 neighbours and $C_{\mathrm{CFL,\,SPH}} = 0.2$ whereas the MFM simulation was run with the M^4 kernel, 32 neighbours, and $C_{\mathrm{CFL,\,MFM}} = 0.1$. We simulated a total of 774,144 gas particles, initially organized in a regular grid (of different spacing in areas A and B). The colour code symbolizes density; only every 70^{th} particle is plotted.

We also stress that our conclusions drawn here apply to pure hydrodynamical applications. In full cosmological runs, many more subroutines come into play, most of which

Solver	$M = 1.5$		$M = 100$	
	M^4	C^6	M^4	C^6
SPH	2:34:28	6:19:55	2:28:33	6:14:36
MFM	3:09:15	8:54:22	3:04:05	8:39:02

Table 5.1: The total runtimes (real time) of all eight shock tube tests in Sec. 5.2. All simulations were run on 32 MPI tasks on a single node with hyperthreading and one OpenMP thread. Each simulation handled 216,090 particles and $t_{\max}$, C_{CFL}, as well as the minimum and maximum allowed time bins have been kept constant. In all cases, the M^4 kernel used 32 neighbours and the C^6 295.

unaffected by the choice of hydro solver. This lessens the impact on total computational time that the chosen CFD scheme has.

The choice of Riemann solver We implemented both the exact Riemann solver by Toro (1999) as well as the approximate HLL solver (Harten et al., 1983), see Sec. 3.3 for a comparison of the two approaches. We carried out a short suit of tests to determine if the trade-off in accuracy when using the HLL solver is worth the gained computation speed.

Tab. 5.2 shows the total runtime of the Sedov blast wave test when performed with the HLL solver, the exact solver, and both compared to SPH. The setup for these tests is the same as described in Sec. 5.3. Throughout all tests, we used $C_{\mathrm{CFL}} = 0.15$. We found the HLL solver to be crashing frequently when using the slope limiting procedure according to Springel (2010) and thus changed to a total variation diminishing (TVD) scalar limiter as in Heß and Springel (2010). This choice leads to a slight increase of diffusion but does not meaningfully affect the runtime on its own.

Solver	M^4	C^6
SPH	0:03:49	0:19:44
MFM (exact)	0:10:04	0:52:41
MFM (HLL)	0:08:54	0:49:39

Table 5.2: Comparison of runtimes (real time) of Sedov blast wave tests in 3D, simulating 64^3 particles. All tests were run on eight MPI ranks.

Here, we see that MFM shows a bigger temporal performance discrepancy to SPH than in the shock tube tests. With equal parameters, MFM takes more than twice as long to execute, independent of the chosen Riemann solver. We note that these tests were executed on a laptop with only moderate resources and suspect these differences to originate in an architecture less optimized towards parallel computing[24]. Tests using only a single task per simulation support this hypothesis as we see a decrease in runtime differences.

[24]Recall that, as mentioned before, MFM needs to perform two additional double loops over particles and their neighbours compared to SPH, therefore also requiring more MPI calls.

The above notwithstanding, we still find that a comparison of SPH with the C^6 kernel to MFM with the M^4 kernel results in a faster computation time by a factor of $\sim$2. Comparing both Riemann solvers, we find that the HLL solver results in a speed-up of 6-13% (for the M^4 and C^6 kernel, respectively). In Fig. 5.10, we contrast this to the qualitative performance of both solvers.

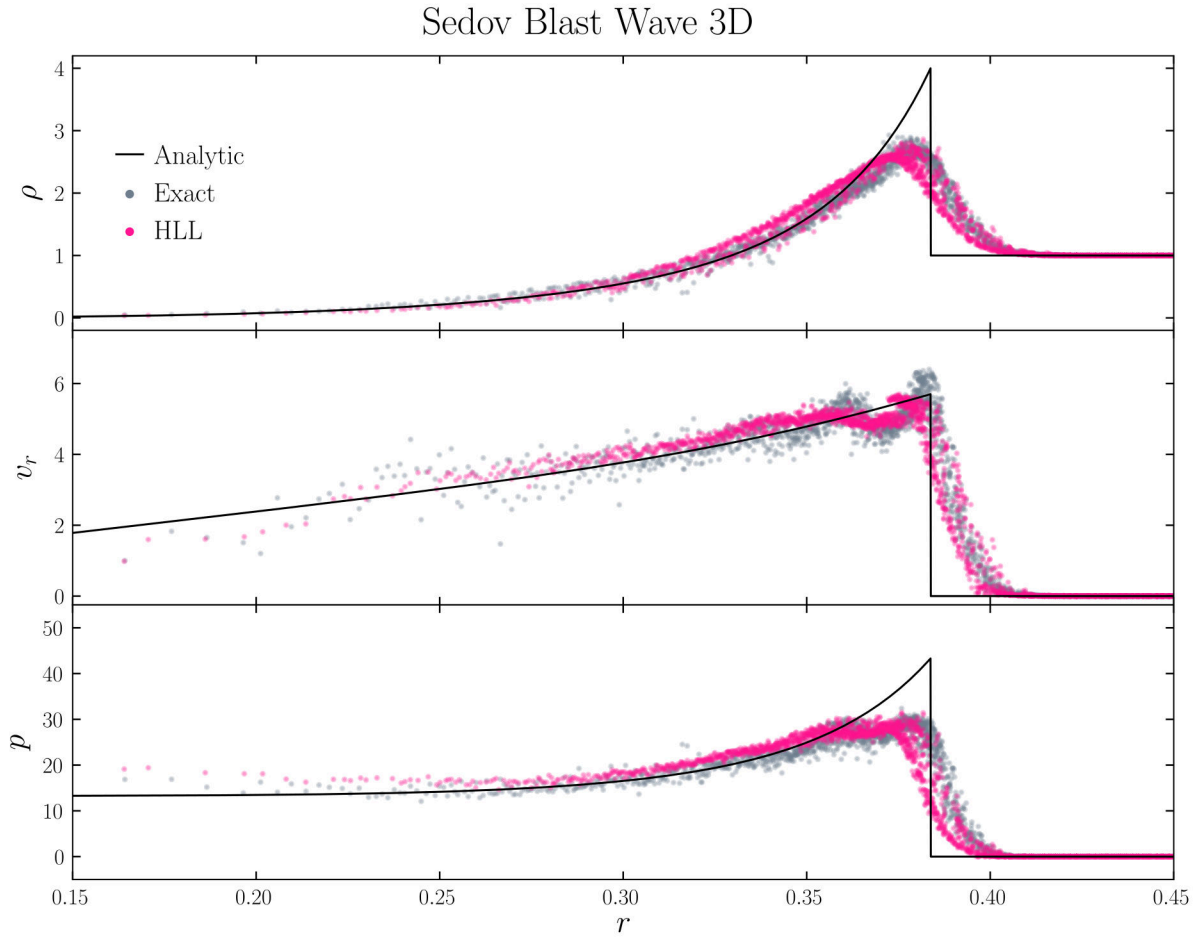


Figure 5.10: Comparison of our two Riemann solvers simulating a Sedov blast wave in 3D. The setup equals the one of Fig. 5.4. The M^4 kernel with 32 neighbours was used in both cases. For the exact solver, we used the slope limiting scheme from Springel (2010) whereas the HLL solver was run alongside the slope limiter of Heß and Springel (2010).

Both Riemann solvers produce similar results. Notable differences include a very slight trailing of the HLL solution (especially distinguishable in the $p(x)$ panel at the bottom) as well as an increase in the noise within the wave preceding the actual shock. These phenomena can be explained by the more diffuse nature of both the HLL solver and the TVD slope limiter. We argue that the two solution agree well enough to justify the choice of the faster scheme. Nevertheless, we found the HLL solver to be less stable than its exact counterpart. As mentioned above, this precluded us from using the slope limiter by Springel (2010). We did not study the interplay of more limiting schemes and Riemann solvers and remark that more optimization would be necessary in this regard.

5.7 Coupling to Gravity

So far, all tests that we performed were simulating hydrodynamical interactions alone. To see if our implementation is of any merit, however, we also ought to ensure that it couples well to other physics modules. While a complete integration of MFM in all additional subroutines of `OpenGadget3` would require considerably more work to be done, we carried out some primary tests to study the interaction of MFM with the N-body gravity solver.

First, we simulate the collapse of a sphere of constant density. Fig. 5.11 shows our results. We see that the initial accelerations due to gravitational forces are resolved equally well by both schemes. The deviations from the ideal line at $r \to 1$ (i.e. at the sphere edges) is a result of the density of the outermost particles being underestimated[25]. Studying the evolution of the half-mass radius of the sphere, we also see good agreement between both methods. Only towards the end of the simulation when the sphere collapse is well advanced, the MFM run produces slightly smaller values of $R_{1/2}$ (i.e. a faster collapse) which are, however, in better agreement with the ideal solution. This shows yet another effect of the suppressed over-smoothing in meshless schemes, postponing the inevitable numerical diffusion of the theoretically ensuing singularity.

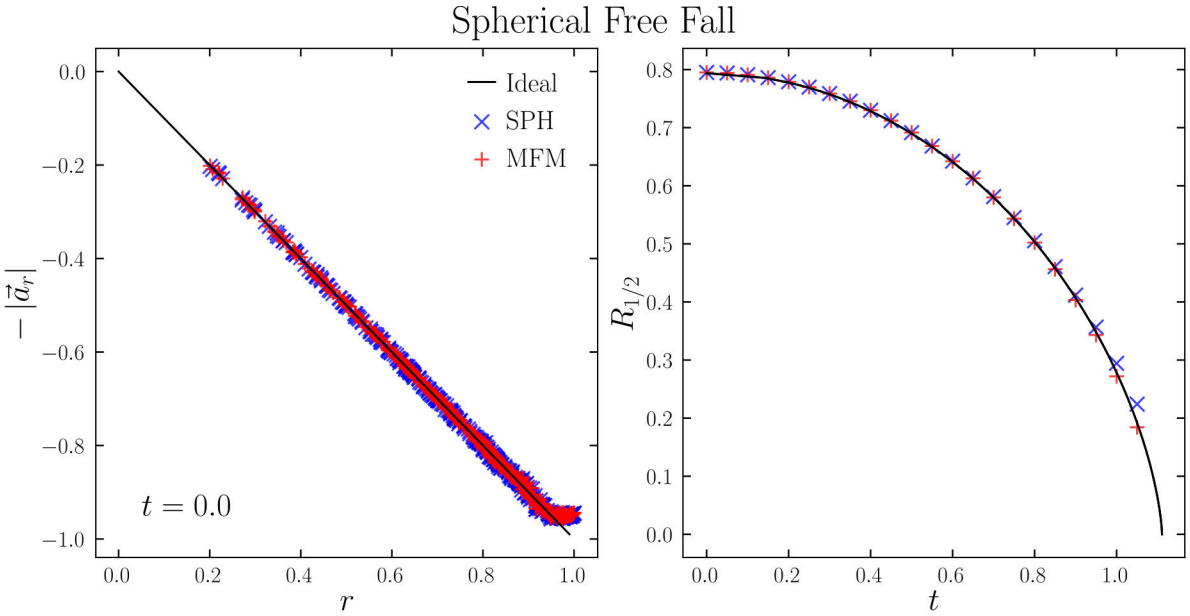


Figure 5.11: Free fall test in 3D using 4,000 particles. The left panel shows the total acceleration $\vec{a}_r$ acting on a particle at the very start of the simulation as a function of distance r to sphere centre. Every tenth particle is plotted. The right panel shows the evolution of the half-mass radius $R_{1/2}$, i.e. the radius containing half of the mass of the total sphere, with time. The MFM simulation used an exact Riemann solver and the TVD slope limiter by Heß and Springel (2010). Both tests were performed with the M^4 kernel and 32 neighbours.

Secondly, we set up a Rayleigh-Taylor instability (see e.g. Sharp 1984). To this extent, we place two fluids in a 2D unit box with a constant gravitational field of $g = -0.5$ in y-direction. We align the fluids such that the heavier one is placed *on top* of the lighter

[25]This, in turn, is a direct consequence of the spherical symmetry and finite extent of the kernel functions. Particles at the sphere edge take the void outside of the sphere into consideration for their density estimate.

one. We apply periodic boundary conditions in x and reflective ones in y. The latter is realized by forcing all particles with $y_i < 0.1$ or $y_i > 0.9$ in place for the entirety of the simulation. To excite an instability, we insert a velocity perturbation at the phase boundary. Self-gravity of gas particles is switched off. The exact set up follows Sec. 4.4.2 in Hopkins (2015).

Fig. 5.12 depicts the outcome of these simulations. Both schemes form two distinct drops that follow the gravitational field and descend with a similar velocity. In SPH, the two drops show significant differences to one another. This is an artefact of small aberrations in the ICs, propagating to different pressure estimates and starting a self-reinforcing pattern. Such an effect is absent in MFM since pressure terms are not directly inferred from density. Other than that, we see phenomena manifest that we already discussed in previous sections: The MFM implementation (once again run with a lot less neighbours) shows considerably more noise and mixing. Whereas the columns above the drops are very sharply defined in SPH, they are smeared out in MFM. The immediate vicinity of the lower ends of the drops also shows more substructure in SPH. We attribute the lack of distinct features in MFM to the relatively low resolution as well as the diffusive slope limiter by Heß and Springel (2010) that was used for its stability.

A more demanding test case we study is that of a hydrostatic sphere. We follow the setup of Viola et al. (2008) and initialize a dark matter (DM) halo with 88,088 particles and a spherical density profile as described in Navarro et al. (1997):

$$\rho(r) = \frac{\rho_0}{\left(r/r_\mathrm{s}\right)\left(1 + r/r_\mathrm{s}\right)^2} \tag{5.5}$$

with central density ρ_0 and scale radius r_s. We afterwards populate this DM halo with 95,156 gas particles such that both components are in hydrostatic equilibrium. Note that this simulation is thus the first in this work to feature more than one type of particle. If all effects are resolved correctly, the initial profile should persist indefinitely. We test our MFM implementation both with fixed and adaptive gravitational softening (AGS).

We show our results in Fig. 5.13. It is apparent that SPH is able to maintain a stable profile (that is, however, slightly different to the initial profile) much more easily. Apart from a somewhat significant difference between the ICs and the first output at $t = 2$, no discernable trend with time is traceable. Note that the small fluctuations near the sphere centre are, in part, due to a low number of particles within each bin combined with a smaller bin size. The MFM simulations, on the other hand, struggle to find an equilibrium configuration. While mass gets accumulated near the sphere centre, internal energy (and thus temperature) is transported outwards. This phenomenon persists when invoking AGS although its amplitude is slightly reduced. Interestingly, however, at late times, we see a turnaround at the centre of the sphere with density decreasing and internal energy increasing again. This, on one hand, could point to a spurious interaction between MFM and AGS not yet fully understood. On the other hand, this might indicate that MFM oscillates around an equilibrium state on massively longer timescales than SPH. Missing viscosity terms and the ensuing lack of energy dissipation might prevent our implementation from relaxing in a timely fashion.

We conclude that our MFM implementation at the current stage is unable to resolve a sphere in hydrostatic equilibrium correctly. While the numerical deviation is not catastrophic, no stable configuration could be reached, even after considerable physical runtime. The reasons for this failure are currently unknown, although we suspect a bug

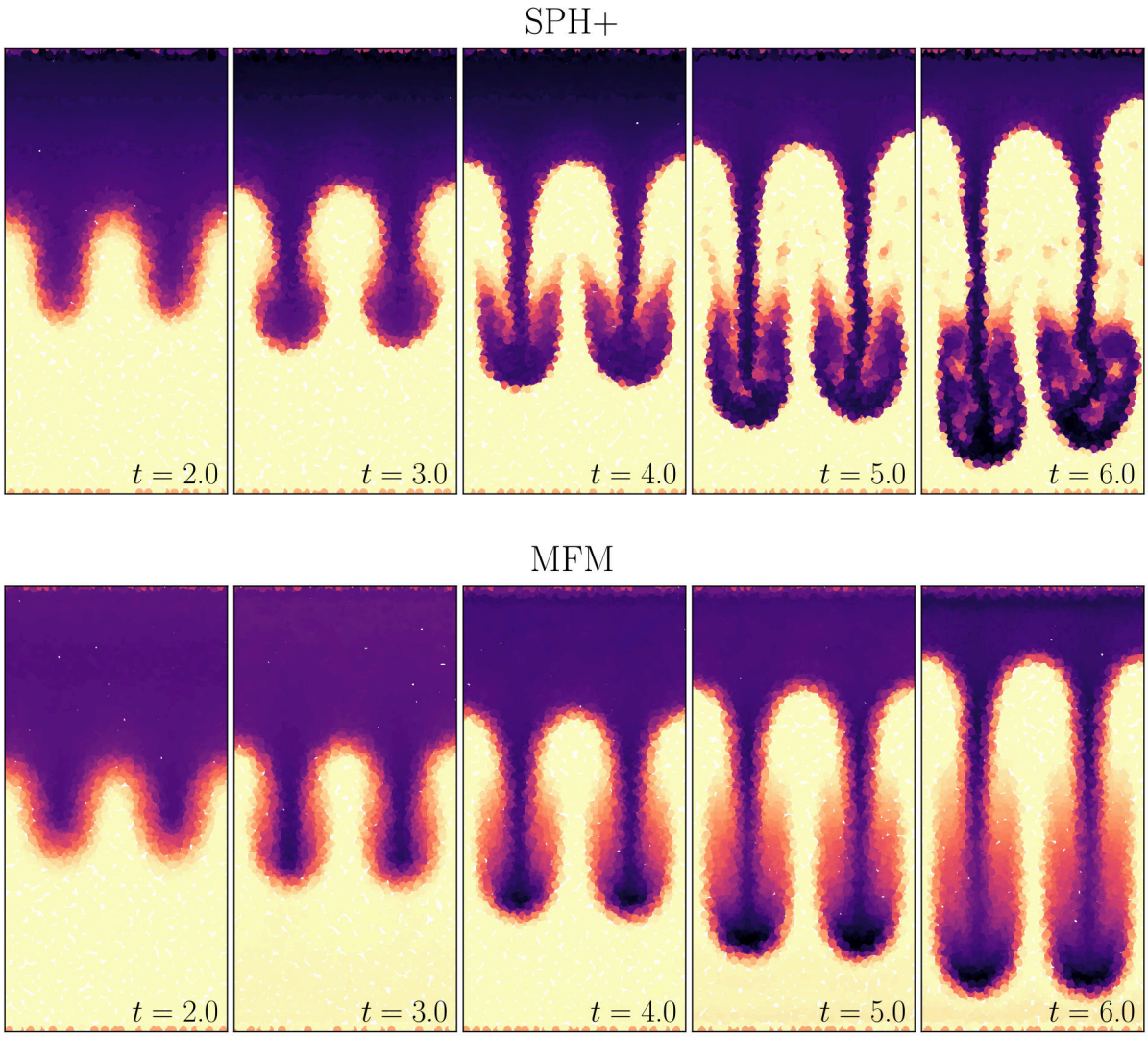


Figure 5.12: Rayleigh-Taylor instability simulated using SPH with all relevant corrective terms (upper panels) and with MFM (lower panels). The simulation followed 65,536 particles; we plotted every seventh of which. Due to the self-similarity of the simulation, we show only half of the box. The colour code indicates density with darker colours corresponding to higher values of ρ.

---

or an oversight in the code to be at fault more so than theoretical limitations. We will shortly revisit the test of the hydrostatic sphere in Sec. 6.2.

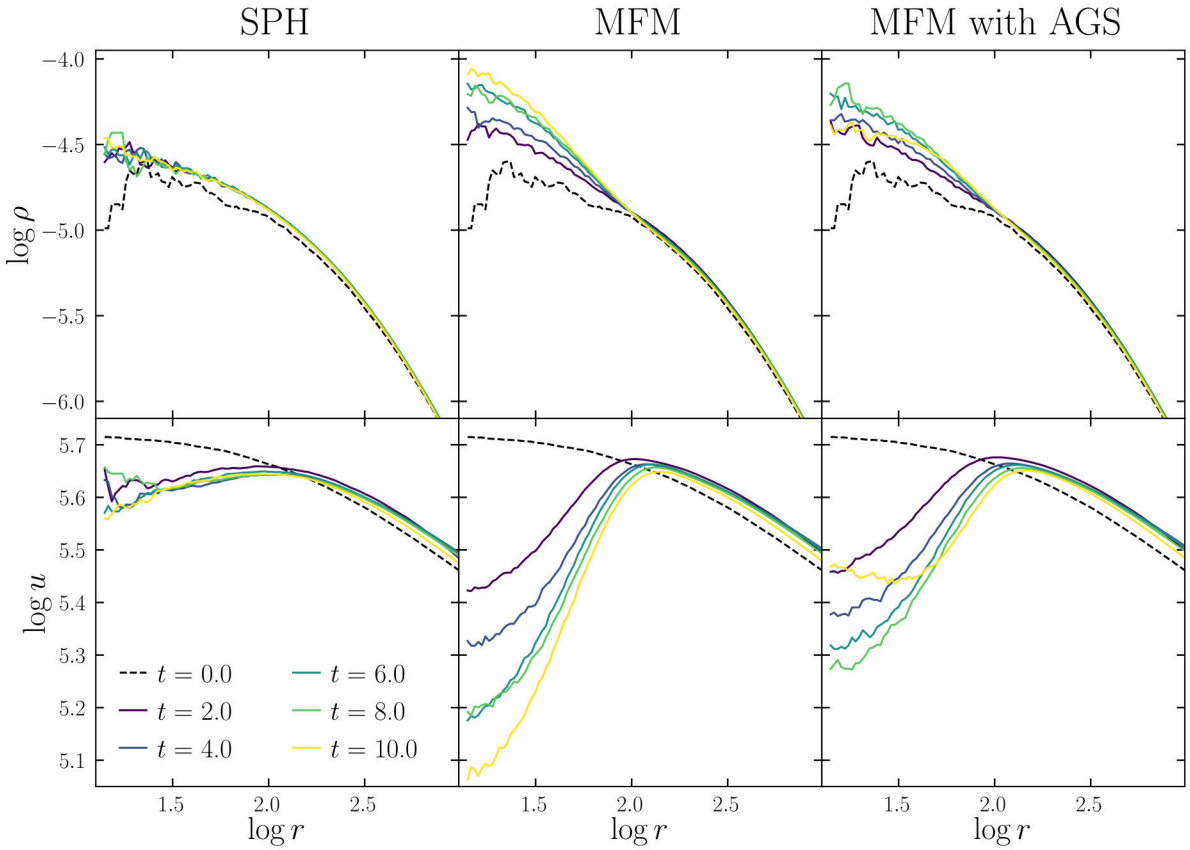


Figure 5.13: Evolution of the hydrostatic sphere. The SPH run used the C^6 kernel, 295 neighbours and $C_{\mathrm{CFL}} = 0.15$; both MFM runs used the M^4 kernel, 32 neighbours and $C_{\mathrm{CFL}} = 0.05$. The first MFM simulation (middle panels) used fixed gravitational softening, as did the SPH simulation (left panels), whereas the right panels show MFM with AGS. $\rho(r)$ is calculated as the sum of particle masses within radius r divided by sphere volume $V(r) = (4/3)\pi r^3$. $u(r)$ is the average internal energy of all particles within r. Both quantities are binned in 100 logarithmically equally sized bins with $r_{\mathrm{min}} = 14$ and $r_{\mathrm{max}} = 1000$. Only gas particles are evaluated.

6 Discussion

While our implementation of an MFM scheme in `OpenGadget3` shows some promising initial results, it is still unfit to be implemented in full cosmological applications. In this section, we will point out the most important aspects that are as of yet missing in our realization and present a brief comparison of our current results to those achieved by other authors.

6.1 Missing Features

- The inclusion of any sub-grid physics was not yet examined. Due to the modular nature of `OpenGadget3`, some of them do not require a lot of (if any) adjustment, such as the black hole model. Others, most importantly the simulation of multiphase particles and chemical networks, ought to be deeply intertwined with the MFM scheme to guarantee a coherent treatment.

- Our implementation is thus far only applicable in Newtonian space, i.e. neglecting dark energy. As pointed out by Hopkins (2015), when switching to a general description in comoving coordinates $\vec{r}_\mathrm{c} = \vec{r}/a(t)$ with the scale factor $a(t)$ (which is `GADGET`'s current approach to tackle cosmic expansion, see Sec. 4.4), the majority of the MFM algorithms remain untouched as long as primitive vectors are converted into physical coordinates before being fed to the Riemann solver. The resulting fluxes are then to be translated back to their comoving counterparts.

- We are lacking any inclusion of magnetic fields in MFM. These would require a considerable expansion of our basic framework, see e.g. Hopkins and Raives (2015) for an exemplary method of integration.

- As described in great detail in Springel (2010) (§ 3.5), Riemann solvers tend to produce erroneous fluxes in regimes where flows are dominated by kinetic energies. Since Riemann solvers in general evolve the total energy (and since this quantity is manifestly conserved), small discretization errors may propagate if $E_\mathrm{kin} \gg E_\mathrm{th}$. Following Bryan et al. (1995), this problem may be alleviated by evolving the internal energy independently in addition to the total energy and using these results if sufficiently supersonic flows are detected.

- While we offer the choice between two Riemann solvers, these are only two extremes out of a wide range of available algorithms: The approach by Toro (1999) is exact at the cost of greater computation time (due to its iterative nature) whereas the HLL solver (Harten et al., 1983) makes a series of assumptions to significantly simplify the problem. However, more advanced approximate schemes (such as the HLLC or HLLE solvers, Toro et al. 1994 and Einfeldt 1988, respectively) might offer a better balance between computation speed and accuracy.

- Similarly, we implemented two slope limiters as well as the possibility to not limit gradients at all. The latter is prone to crashes since no safeguard prevents e.g. too much energy being transported from one particle to another, leading to negative (and therefore unphysical) energy terms. However, we also experienced instabilities with the implemented limiters. While we found the approach by Heß and Springel (2010) to be generally more stable than the one by Springel (2010), other limiters might prove to be even more robust.

- At the moment, the second-order accuracy of the gradient estimates is hard-coded into our approach even though arbitrarily high-order schemes are possible. When pertaining the locally-centred least-square approach, this is realized by increasing the size of the $\underline{\underline{B_i}}$ and $\underline{\underline{E_i}}$ matrices (see Sec. 3.2.2). Since $\underline{\underline{B_i}}$ is recovered from $\underline{\underline{E_i}}$ through inversion, this also requires more CPU time. In-depth numerical analysis would be necessary to verify if the second-order approach truly is the best compromise (as claimed e.g. by Hopkins 2015).

6.2 Comparison to Other Codes

The pure hydrodynamical treatment in our code is comparable in quality to that of other codes that feature an MFM implementation.

In Fig. 6.1, we show Sod shock tubes simulated with the `GANDALF` (Hubber et al., 2018) code on which our own implementation is based. Unsurprisingly, we observe similar artefacts — MFM generally leads to less over-smoothing but, as the contact discontinuity in the $u(x)$ plot shows (lower panels), the effects of the inconsistent treatment of discontinuities remain. Note that Fig. 6.1 also depicts simulations using unequal particle masses which produce rather spurious results in MFM. Based on those findings we did not attempt such runs.

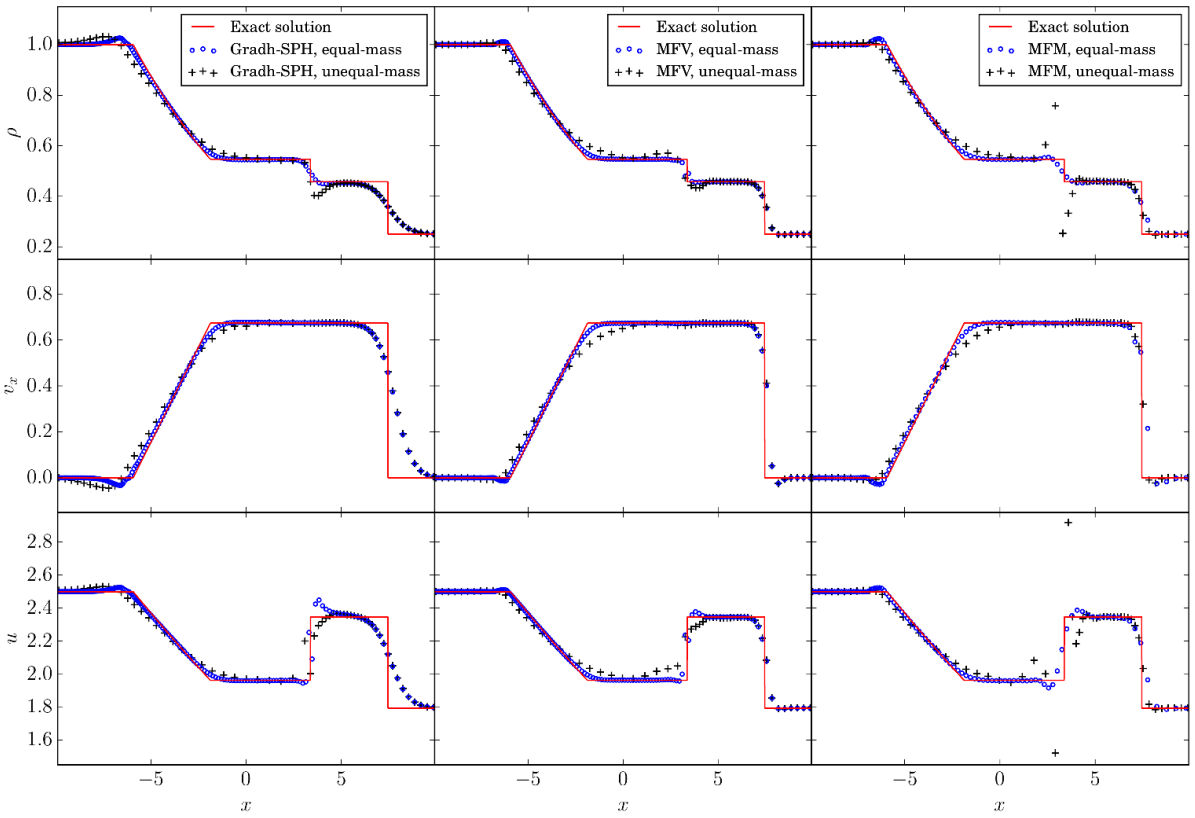


Figure 6.1: Sod shock tubes as simulated with `GANDALF`. *Gradh-SPH* refers to the standard SPH approach with the ∇h correction terms discussed in Sec. 2.2.5. The unequal-mass particles were initialized equally spaced; taken from Hubber et al. (2018).

In Fig. 3.6, we already showed the Kelvin-Helmholtz instability as performed with the `GIZMO` code (Hopkins, 2015). Their findings generally resemble ours (see Fig. 5.9)

with some notable differences: Comparing the solutions obtained with SPH, we clearly see a better resolution of the instability and the ensuing mixing in our approach (when using correction terms). This is mainly attributed to the considerably higher neighbour number we used but shows that modern SPH with correctly chosen parameters is indeed capable of resolving fluid mixing to a certain degree. Secondly, our MFM approach appears to be slightly more diffusive than the one in `GIZMO`. While we did not carry out an in-depth analysis to establish the reason for this, we note that a fair amount of parameters can — directly or indirectly — impact the noise and diffusivity of the scheme. Among those are C_{CFL}, the allowed deviation of neighbour number when determining h (see Eq. 2.26), as well as the chosen Riemann solver and slope limiter.

A comparison of the Rayleigh-Taylor instability (see Fig. 5.12) in our code and in `GIZMO` yields a similar verdict. While the coupling to gravity shows no significant differences, our MFM scheme is more diffusive, producing less substructure. In this case, in addition to the possible sources mentioned above, our simulation also used a considerably lower number of particles (by a factor of 16), naturally decreasing the resolution further.

No other authors — to the best of our knowledge — published an attempt to resolve the hydrostatic sphere with MFM. We therefore repeated our simulation (see Sec. 5.7) with the publicly available version of `GIZMO` for comparison. We used the same ICs file and parameters, as well as utilizing the AGS mode of `GIZMO`. For consistency, we also repeated the simulation with `OpenGadget3`'s SPH solver and AGS enabled. Fig. 6.2 depicts the outcome of these runs. It is evident that AGS also improves the results for SPH with both $\rho(r)$ and $u(r)$ resembling the initial profile more closely. Only for large r we see a slight overshoot of internal energy. Comparing the results of MFM obtained with our implementation and `GIZMO` we find the latter to perform much better. The calculated profiles maintain similar shapes to the ICs even at late times. One peculiarity we point out is the $u(r)$ line at $t = 8$, following the ICs almost exactly for small radii but then converging to the relatively stable solution at later times for greater values of r. This indicates that, also in `GIZMO`, fluctuations of the results remain, albeit significantly less pronounced than with our MFM implementation.

We note that we also attempted to simulate an Evrard sphere collapse (Evrard, 1988) as done by Hopkins (2015). However, our attempts to do so were unfruitful since we encountered numerical instabilities. While we are, at the time of this writing, unable to name the reason for this with certainty, we suspect either the missing possibility to evolve the internal energy independently from the total energy (see Sec. 6.1) or a wrong choice of slope limiter to be responsible. Whatever the cause, it might be expected that the solution may also further improve the outcome of some of the tests we showed in this work.

To conclude, we find good agreement of our implementation with other codes concerning pure hydrodynamical interactions with the only difference being enhanced noise in our scheme. For advanced gravitational tests, however, we see considerable discrepancies. Compared to `GIZMO`, our implementation lacks the HLLC Riemann solver as well as the slope limiting procedure described by Hopkins (2015), both being invoked by default in `GIZMO`. An implementation of these two might prove useful in determining the root of the current limitations.

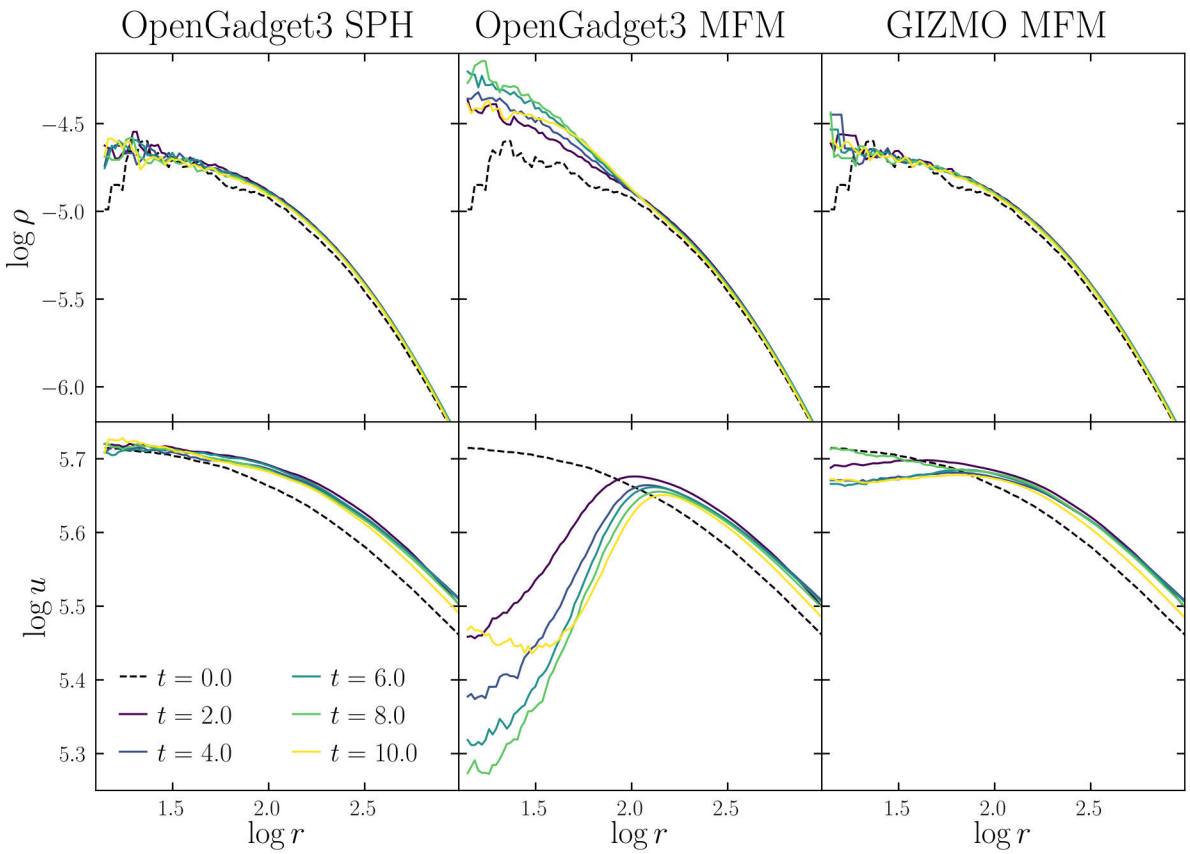


Figure 6.2: Evolution of the hydrostatic sphere as simulated with `OpenGadget3` and `GIZMO`. All runs used AGS and the SPH simulation used $C_{\mathrm{CFL}} = 0.1$. All remaining parameters are the same as described in Sec. 5.7. The two central panels depict the same run as the two right panels in Fig. 5.13 (note the slightly adjusted y-axes).

7 Conclusion and Outlook

We implemented an MFM scheme in the cosmological code `OpenGadget3`, following the realization of Hubber et al. (2018). While the installation at the current stage is not yet fit for full cosmological simulations, we investigated its performance on common test cases in great detail and found very promising results.

For pure hydrodynamical runs, we were able to achieve outcomes that are of the same quality as those obtained using a modern SPH solver. More importantly, we demonstrated that MFM can be operated with both a simpler kernel function and a lower number of neighbours compared to SPH, halving the runtime and lessening various over-smoothing effects. However, we also found our solutions with MFM to be consistently more noisy. We therefore recommend to reduce the Courant-Friedrichs-Lewy number when using MFM in order to get a better temporal resolution. We furthermore showed the necessity of a wake up scheme to avoid spurious particle penetration.

We coupled our implementation to the N-body gravity solver of `OpenGadget3`. For simple test cases, we found good agreement with the results achieved with SPH, in part trumping them. More involved tests, however, revealed our code to struggle in terms of both physical and numerical stability. Since said test cases (namely the hydrostatic and the Evrard sphere) are resolved well with other MFM codes, we suspect the issue to originate in an implementation error rather than a true limit of the scheme.

For future work, finding the root of this problem surely would benefit the performance of our MFM scheme greatly. Likewise, many more features (such as coupling to additional physics modules or more advanced slope limiters) are as of yet missing and would further enhance the capabilities and utilities of the implementation. Ultimately, we would like to apply MFM to cosmological simulations in order to study in detail how its results differ from those achieved with SPH. We remain positive that with a stable version and a better investigated parameter space MFM might provide a way to solve CFD both quicker and more accurately. While currently, not many cosmological codes offer such an approach, we expect this to change in upcoming years.

Bibliography

Agertz, O., Moore, B., Stadel, J., Potter, D., Miniati, F., Read, J., Mayer, L., Gawryszczak, A., Kravtsov, A., Nordlund, Å., Pearce, F., Quilis, V., Rudd, D., Springel, V., Stone, J., Tasker, E., Teyssier, R., Wadsley, J., and Walder, R. (2007). Fundamental differences between SPH and grid methods. MNRAS, 380(3):963–978.

Arth, A., Dolag, K., Beck, A. M., Petkova, M., and Lesch, H. (2014). Anisotropic thermal conduction in galaxy clusters with MHD in Gadget. *arXiv e-prints*, page arXiv:1412.6533.

Barnes, J. and Hut, P. (1986). A hierarchical O(N log N) force-calculation algorithm. Nature, 324(6096):446–449.

Beck, A. M., Murante, G., Arth, A., Remus, R. S., Teklu, A. F., Donnert, J. M. F., Planelles, S., Beck, M. C., Förster, P., Imgrund, M., Dolag, K., and Borgani, S. (2016). An improved SPH scheme for cosmological simulations. MNRAS, 455(2):2110–2130.

Berger, M. and Oliger, J. (1984). Adaptive mesh refinement for hyperbolic partial differential equations. *Journal of Computational Physics*, 53(3):484–512.

Berger, M. J. and Colella, P. (1989). Local Adaptive Mesh Refinement for Shock Hydrodynamics. *Journal of Computational Physics*, 82(1):64–84.

Bondi, H. (1952). On spherically symmetrical accretion. MNRAS, 112:195.

Book, D. L. (1994). The Sedov self-similar point blast solutions in nonuniform media. *Shock Waves*, 4(1):1–10.

Borgani, S., Murante, G., Springel, V., Diaferio, A., Dolag, K., Moscardini, L., Tormen, G., Tornatore, L., and Tozzi, P. (2004). X-ray properties of galaxy clusters and groups from a cosmological hydrodynamical simulation. MNRAS, 348(3):1078–1096.

Børve, S., Omang, M., and Trulsen, J. (2001). Regularized Smoothed Particle Hydrodynamics: A New Approach to Simulating Magnetohydrodynamic Shocks. ApJ, 561(1):82–93.

Brandenburg, A. and Subramanian, K. (2005). Astrophysical magnetic fields and nonlinear dynamo theory. Phys. Rep., 417(1-4):1–209.

Bryan, G. L., Norman, M. L., O'Shea, B. W., Abel, T., Wise, J. H., Turk, M. J., Reynolds, D. R., Collins, D. C., Wang, P., Skillman, S. W., Smith, B., Harkness, R. P., Bordner, J., Kim, J.-h., Kuhlen, M., Xu, H., Goldbaum, N., Hummels, C., Kritsuk, A. G., Tasker, E., Skory, S., Simpson, C. M., Hahn, O., Oishi, J. S., So, G. C., Zhao, F., Cen, R., Li, Y., and The Enzo Collaboration (2014). ENZO: An Adaptive Mesh Refinement Code for Astrophysics. ApJS, 211:19.

Bryan, G. L., Norman, M. L., Stone, J. M., Cen, R., and Ostriker, J. P. (1995). A piecewise parabolic method for cosmological hydrodynamics. *Computer Physics Communications*, 89(1-3):149–168.

Böss, L. et al. (in prep.). Unpublished.

Chabrier, G. (2003). Galactic Stellar and Substellar Initial Mass Function. PASP, 115(809):763–795.

Chorin, A. J. (1976). Random choice solution of hyperbolic systems. *J. Comp. Phys*, pages 517–533.

Crain, R. A., Schaye, J., Bower, R. G., Furlong, M., Schaller, M., Theuns, T., Dalla Vecchia, C., Frenk, C. S., McCarthy, I. G., Helly, J. C., Jenkins, A., Rosas-Guevara, Y. M., White, S. D. M., and Trayford, J. W. (2015). The EAGLE simulations of galaxy formation: calibration of subgrid physics and model variations. MNRAS, 450(2):1937–1961.

Davé, R., Anglés-Alcázar, D., Narayanan, D., Li, Q., Rafieferantsoa, M. H., and Appleby, S. (2019). SIMBA: Cosmological simulations with black hole growth and feedback. MNRAS, 486(2):2827–2849.

Davis, S. F. (1988). Simplified second-order godunov-type methods. *SIAM J. Sci. Stat. Comput.*, 9(3):445–473.

Dehnen, W. and Aly, H. (2012). Improving convergence in smoothed particle hydrodynamics simulations without pairing instability. *Monthly Notices of the Royal Astronomical Society*, 425.

Dolag, K., Komatsu, E., and Sunyaev, R. (2016). SZ effects in the Magneticum Pathfinder simulation: comparison with the Planck, SPT, and ACT results. MNRAS, 463(2):1797–1811.

Dolag, K. and Stasyszyn, F. (2009). An MHD GADGET for cosmological simulations. MNRAS, 398(4):1678–1697.

Einfeldt, B. (1988). On godunov-type methods for gas dynamics. *Siam Journal on Numerical Analysis - SIAM J NUMER ANAL*, 25:294–318.

Evrard, A. E. (1988). Beyond N-body: 3D cosmological gas dynamics. MNRAS, 235:911–934.

Fabjan, D., Borgani, S., Tornatore, L., Saro, A., Murante, G., and Dolag, K. (2010). Simulating the effect of active galactic nuclei feedback on the metal enrichment of galaxy clusters. MNRAS, 401(3):1670–1690.

Felker, K. G. and Stone, J. M. (2018). A fourth-order accurate finite volume method for ideal MHD via upwind constrained transport. *Journal of Computational Physics*, 375:1365–1400.

Fromang, S., Papaloizou, J., Lesur, G., and Heinemann, T. (2007). MHD simulations of the magnetorotational instability in a shearing box with zero net flux. II. The effect of transport coefficients. A&A, 476(3):1123–1132.

Gaburov, E. and Nitadori, K. (2011). Astrophysical weighted particle magnetohydrodynamics. MNRAS, 414(1):129–154.

Gingold, R. A. and Monaghan, J. J. (1977). Smoothed particle hydrodynamics: theory and application to non-spherical stars. MNRAS, 181:375–389.

Godunov, S. K. and Bohachevsky, I. (1959). Finite difference method for numerical computation of discontinuous solutions of the equations of fluid dynamics. *Matematičeskij sbornik*, 47(89)(3):271–306.

Grenier, I. A., Black, J. H., and Strong, A. W. (2015). The Nine Lives of Cosmic Rays in Galaxies. ARA&A, 53:199–246.

Harten, A., Lax, P., and van Leer, B. (1983). On upstream differencing and godunov-type schemes for hyperbolic conservation laws. *SIAM Rev*, 25:35–61.

Hernquist, L. (1993). Some Cautionary Remarks about Smoothed Particle Hydrodynamics. ApJ, 404:717.

Heß, S. and Springel, V. (2010). Particle hydrodynamics with tessellation techniques. MNRAS, 406(4):2289–2311.

Hirschmann, M., Dolag, K., Saro, A., Bachmann, L., Borgani, S., and Burkert, A. (2014). Cosmological simulations of black hole growth: AGN luminosities and downsizing. MNRAS, 442(3):2304–2324.

Hopkins, P. F. (2013). A general class of Lagrangian smoothed particle hydrodynamics methods and implications for fluid mixing problems. MNRAS, 428(4):2840–2856.

Hopkins, P. F. (2015). A new class of accurate, mesh-free hydrodynamic simulation methods. *Monthly Notices of the Royal Astronomical Society*, 450(1):53–110.

Hopkins, P. F. (2017). A New Public Release of the GIZMO Code. *arXiv e-prints*, page arXiv:1712.01294.

Hopkins, P. F. and Raives, M. J. (2015). Accurate, meshless methods for magnetohydrodynamics. *Monthly Notices of the Royal Astronomical Society*, 455(1):51–88.

Hubber, D. A., Rosotti, G. P., and Booth, R. A. (2018). GANDALF - Graphical Astrophysics code for N-body Dynamics And Lagrangian Fluids. MNRAS, 473(2):1603–1632.

Iannuzzi, F. and Dolag, K. (2011). Adaptive gravitational softening in GADGET. MNRAS, 417(4):2846–2859.

Jalil, M., Asghar, S., and Yasmeen, S. (2017). An exact solution of mhd boundary layer flow of dusty fluid over a stretching surface. *Mathematical Problems in Engineering*, 2017:1–5.

Käppeli, R. and Mishra, S. (2014). *Structure Preserving Schemes*, volume 488 of *Astronomical Society of the Pacific Conference Series*, page 231.

Katz, A. and Sankaran, V. (2011). Mesh quality effects on the accuracy of CFD solutions on unstructured meshes. *Journal of Computational Physics*, 230(20):7670–7686.

Kennicutt, Robert C., J. (1989). The Star Formation Law in Galactic Disks. ApJ, 344:685.

Kereš, D., Katz, N., Fardal, M., Davé, R., and Weinberg, D. H. (2009). Galaxies in a simulated ΛCDM Universe - I. Cold mode and hot cores. MNRAS, 395(1):160–179.

Kroupa, P. (2001). On the variation of the initial mass function. MNRAS, 322(2):231–246.

Ladonkina, M. E. and Tishkin, V. F. (2015). Godunov method: a generalization using piecewise polynomial approximations. *Differential Equations*, 51:895–903.

Landau, L. and Lifshitz, E. (2013). *Fluid Mechanics*. Number Bd. 6. Elsevier Science.

Lanson, N. and Vila, J.-P. (2008a). Renormalized meshfree schemes i: Consistency, stability, and hybrid methods for conservation laws. *SIAM Journal on Numerical Analysis*, 46(4):1912–1934.

Lanson, N. and Vila, J. P. (2008b). Renormalized meshfree schemes ii: Convergence for scalar conservation laws. *SIAM J. Numerical Analysis*, 46:1935–1964.

Lebedeva, N. and Osiptsov, A. (2016). A combined lagrangian method for simulation of axisymmetric gas-particle vortex flows. *Fluid Dynamics*, 51:647–659.

Lee, S.-Y., Ziebell, L. F., Yoon, P. H., Gaelzer, R., and Lee, E. S. (2019). Particle-in-cell and Weak Turbulence Simulations of Plasma Emission. ApJ, 871(1):74.

Lerche, I. (1967). Unstable Magnetosonic Waves in a Relativistic Plasma. ApJ, 147:689.

Lucy, L. B. (1977). A numerical approach to the testing of the fission hypothesis. AJ, 82:1013–1024.

Luo, H., Baum, J. D., and Löhner, R. (2008). A discontinuous galerkin method based on a taylor basis for the compressible flows on arbitrary grids. *Journal of Computational Physics*, 227(20):8875 – 8893.

Mihalas, D. and Mihalas, B. W. (1984). *Foundations of radiation hydrodynamics*.

Miniati, F. (2001). COSMOCR: A numerical code for cosmic ray studies in computational cosmology. *Computer Physics Communications*, 141(1):17–38.

Mishchenko, A. V., Godenko, E. A., and Izmodenov, V. V. (2020). Lagrangian fluid approach for the modelling of peculiarities of the interstellar dust distribution in the astrospheres/heliosphere. MNRAS, 491(2):2808–2821.

Monaghan, J. J. (1992). Smoothed particle hydrodynamics. ARA&A, 30:543–574.

Monaghan, J. J. (1997). SPH and Riemann Solvers. *Journal of Computational Physics*, 136(2):298–307.

Murante, G., Monaco, P., Giovalli, M., Borgani, S., and Diaferio, A. (2010). A subresolution multiphase interstellar medium model of star formation and supernova energy feedback. MNRAS, 405(3):1491–1512.

Navarro, J. F., Frenk, C. S., and White, S. D. M. (1997). A Universal Density Profile from Hierarchical Clustering. ApJ, 490(2):493–508.

Okamoto, T., Jenkins, A., Eke, V. R., Quilis, V., and Frenk, C. S. (2003). Momentum transfer across shear flows in smoothed particle hydrodynamic simulations of galaxy formation. MNRAS, 345(2):429–446.

Padovani, P. and Matteucci, F. (1993). Stellar Mass Loss in Elliptical Galaxies and the Fueling of Active Galactic Nuclei. ApJ, 416:26.

Park, J., Ren, C., Workman, J. C., and Blackman, E. G. (2013). Particle-in-cell Simulations of Particle Energization via Shock Drift Acceleration from Low Mach Number Quasi-perpendicular Shocks in Solar Flares. ApJ, 765(2):147.

Parker, E. N. (1958). Dynamics of the Interplanetary Gas and Magnetic Fields. ApJ, 128:664.

Petkova, M. and Springel, V. (2009). An implementation of radiative transfer in the cosmological simulation code GADGET. MNRAS, 396(3):1383–1403.

Pfrommer, C., Springel, V., Enßlin, T. A., and Jubelgas, M. (2006). Detecting shock waves in cosmological smoothed particle hydrodynamics simulations. MNRAS, 367(1):113–131.

Phillips, G. J. and Monaghan, J. J. (1985). A numerical method for three-dimensional simulations of collapsing, isothermal, magnetic gas clouds. MNRAS, 216:883–895.

Planck Collaboration, Ade, P. A. R., Aghanim, N., Arnaud, M., Ashdown, M., Aumont, J., Baccigalupi, C., Banday, A. J., Barreiro, R. B., Bartlett, J. G., Bartolo, N., Battaner, E., Battye, R., Benabed, K., Benoît, A., Benoit-Lévy, A., Bernard, J. P., Bersanelli, M., Bielewicz, P., Bock, J. J., Bonaldi, A., Bonavera, L., Bond, J. R., Borrill, J., Bouchet, F. R., Boulanger, F., Bucher, M., Burigana, C., Butler, R. C., Calabrese, E., Cardoso, J. F., Catalano, A., Challinor, A., Chamballu, A., Chary, R. R., Chiang, H. C., Chluba, J., Christensen, P. R., Church, S., Clements, D. L., Colombi, S., Colombo, L. P. L., Combet, C., Coulais, A., Crill, B. P., Curto, A., Cuttaia, F., Danese, L., Davies, R. D., Davis, R. J., de Bernardis, P., de Rosa, A., de Zotti, G., Delabrouille, J., Désert, F. X., Di Valentino, E., Dickinson, C., Diego, J. M., Dolag, K., Dole, H., Donzelli, S., Doré, O., Douspis, M., Ducout, A., Dunkley, J., Dupac, X., Efstathiou, G., Elsner, F., Enßlin, T. A., Eriksen, H. K., Farhang, M., Fergusson, J., Finelli, F., Forni, O., Frailis, M., Fraisse, A. A., Franceschi, E., Frejsel, A., Galeotta, S., Galli, S., Ganga, K., Gauthier, C., Gerbino, M., Ghosh, T., Giard, M., Giraud-Héraud, Y., Giusarma, E., Gjerløw, E., González-Nuevo, J., Górski, K. M., Gratton, S., Gregorio, A., Gruppuso, A., Gudmundsson, J. E., Hamann, J., Hansen, F. K., Hanson, D., Harrison, D. L., Helou, G., Henrot-Versillé, S., Hernández-Monteagudo, C., Herranz, D., Hildebrand t, S. R., Hivon, E., Hobson, M., Holmes, W. A., Hornstrup, A., Hovest, W., Huang, Z., Huffenberger, K. M., Hurier, G., Jaffe, A. H., Jaffe, T. R., Jones, W. C., Juvela, M., Keihänen, E., Keskitalo, R., Kisner, T. S., Kneissl, R., Knoche, J., Knox, L., Kunz, M., Kurki-Suonio, H., Lagache, G., Lähteenmäki, A., Lamarre, J. M., Lasenby, A., Lattanzi, M., Lawrence, C. R., Leahy, J. P., Leonardi, R., Lesgourgues, J., Levrier, F., Lewis, A., Liguori, M., Lilje, P. B., Linden-Vørnle, M., López-Caniego, M., Lubin, P. M., Macías-Pérez, J. F., Maggio, G., Maino, D., Mandolesi, N., Mangilli, A., Marchini, A., Maris, M., Martin, P. G., Martinelli, M., Martínez-González, E., Masi, S., Matarrese, S., McGehee, P., Meinhold, P. R., Melchiorri, A., Melin, J. B., Mendes, L., Mennella, A., Migliaccio, M., Millea, M., Mitra, S., Miville-Deschênes, M. A., Moneti, A., Montier, L., Morgante, G., Mortlock, D., Moss, A., Munshi, D., Murphy, J. A., Naselsky, P., Nati, F., Natoli, P., Netterfield, C. B., Nørgaard-Nielsen, H. U., Noviello, F., Novikov, D., Novikov, I., Oxborrow, C. A., Paci, F., Pagano, L., Pajot, F., Paladini, R., Paoletti, D., Partridge, B., Pasian, F., Patanchon, G., Pearson, T. J., Perdereau, O., Perotto, L., Perrotta, F., Pettorino, V., Piacentini, F., Piat, M., Pierpaoli, E., Pietrobon, D., Plaszczynski, S., Pointecouteau, E., Polenta, G., Popa, L.,

Pratt, G. W., Prézeau, G., Prunet, S., Puget, J. L., Rachen, J. P., Reach, W. T., Rebolo, R., Reinecke, M., Remazeilles, M., Renault, C., Renzi, A., Ristorcelli, I., Rocha, G., Rosset, C., Rossetti, M., Roudier, G., Rouillé d'Orfeuil, B., Rowan-Robinson, M., Rubiño-Martín, J. A., Rusholme, B., Said, N., Salvatelli, V., Salvati, L., Sandri, M., Santos, D., Savelainen, M., Savini, G., Scott, D., Seiffert, M. D., Serra, P., Shellard, E. P. S., Spencer, L. D., Spinelli, M., Stolyarov, V., Stompor, R., Sudiwala, R., Sunyaev, R., Sutton, D., Suur-Uski, A. S., Sygnet, J. F., Tauber, J. A., Terenzi, L., Toffolatti, L., Tomasi, M., Tristram, M., Trombetti, T., Tucci, M., Tuovinen, J., Türler, M., Umana, G., Valenziano, L., Valiviita, J., Van Tent, F., Vielva, P., Villa, F., Wade, L. A., Wandelt, B. D., Wehus, I. K., White, M., White, S. D. M., Wilkinson, A., Yvon, D., Zacchei, A., and Zonca, A. (2016). Planck 2015 results. XIII. Cosmological parameters. A&A, 594:A13.

Price, D. J. (2008). Modelling discontinuities and Kelvin Helmholtz instabilities in SPH. *Journal of Computational Physics*, 227(24):10040–10057.

Price, D. J. (2012). Smoothed particle hydrodynamics and magnetohydrodynamics. *Journal of Computational Physics*, 231(3):759–794.

Price, D. J. and Monaghan, J. J. (2004). Smoothed Particle Magnetohydrodynamics - II. Variational principles and variable smoothing-length terms. MNRAS, 348(1):139–152.

Ragagnin, A., Dolag, K., Biffi, V., Cadolle Bel, M., Hammer, N. J., Krukau, A., Petkova, M., and Steinborn, D. (2017). A web portal for hydrodynamical, cosmological simulations. *Astronomy and Computing*, 20:52–67.

Read, J. I. and Hayfield, T. (2012). SPHS: smoothed particle hydrodynamics with a higher order dissipation switch. MNRAS, 422(4):3037–3055.

Rosswog, S. (2015). Boosting the accuracy of SPH techniques: Newtonian and special-relativistic tests. MNRAS, 448(4):3628–3664.

Saitoh, T. R. and Makino, J. (2009). A Necessary Condition for Individual Time Steps in SPH Simulations. ApJ, 697(2):L99–L102.

Salpeter, E. E. (1955). The Luminosity Function and Stellar Evolution. ApJ, 121:161.

Schaye, J., Crain, R. A., Bower, R. G., Furlong, M., Schaller, M., Theuns, T., Dalla Vecchia, C., Frenk, C. S., McCarthy, I. G., Helly, J. C., Jenkins, A., Rosas-Guevara, Y. M., White, S. D. M., Baes, M., Booth, C. M., Camps, P., Navarro, J. F., Qu, Y., Rahmati, A., Sawala, T., Thomas, P. A., and Trayford, J. (2015). The EAGLE project: simulating the evolution and assembly of galaxies and their environments. MNRAS, 446(1):521–554.

Schaye, J., Dalla Vecchia, C., Booth, C. M., Wiersma, R. P. C., Theuns, T., Haas, M. R., Bertone, S., Duffy, A. R., McCarthy, I. G., and van de Voort, F. (2010). The physics driving the cosmic star formation history. MNRAS, 402(3):1536–1560.

Schmidt, M. (1959). The Rate of Star Formation. ApJ, 129:243.

Schulze, F., Remus, R.-S., Dolag, K., Burkert, A., Emsellem, E., and van de Ven, G. (2018). Kinematics of simulated galaxies - I. Connecting dynamical and morphological properties of early-type galaxies at different redshifts. MNRAS, 480(4):4636–4658.

Sedov, L. I. (1959). *Similarity and Dimensional Methods in Mechanics*.

Sharp, D. H. (1984). An overview of Rayleigh-Taylor instability. *Physica D Nonlinear Phenomena*, 12(1):3,IN1,11–10,IN10,18.

Skillman, S. W., O'Shea, B. W., Hallman, E. J., Burns, J. O., and Norman, M. L. (2008). Cosmological Shocks in Adaptive Mesh Refinement Simulations and the Acceleration of Cosmic Rays. ApJ, 689(2):1063–1077.

Sod, G. A. (1978). Review. A Survey of Several Finite Difference Methods for Systems of Nonlinear Hyperbolic Conservation Laws. *Journal of Computational Physics*, 27(1):1–31.

Springel, V. (2005). The cosmological simulation code GADGET-2. MNRAS, 364(4):1105–1134.

Springel, V. (2010). E pur si muove: Galilean-invariant cosmological hydrodynamical simulations on a moving mesh. MNRAS, 401(2):791–851.

Springel, V., Di Matteo, T., and Hernquist, L. (2005a). Modelling feedback from stars and black holes in galaxy mergers. MNRAS, 361(3):776–794.

Springel, V. and Hernquist, L. (2002). Cosmological smoothed particle hydrodynamics simulations: the entropy equation. MNRAS, 333(3):649–664.

Springel, V. and Hernquist, L. (2003). Cosmological smoothed particle hydrodynamics simulations: a hybrid multiphase model for star formation. MNRAS, 339(2):289–311.

Springel, V., White, S. D. M., Jenkins, A., Frenk, C. S., Yoshida, N., Gao, L., Navarro, J., Thacker, R., Croton, D., Helly, J., Peacock, J. A., Cole, S., Thomas, P., Couchman, H., Evrard, A., Colberg, J., and Pearce, F. (2005b). Simulations of the formation, evolution and clustering of galaxies and quasars. Nature, 435(7042):629–636.

Springel, V., Yoshida, N., and White, S. D. M. (2001). GADGET: a code for collisionless and gasdynamical cosmological simulations. New A, 6(2):79–117.

Steinborn, L. K., Dolag, K., Comerford, J. M., Hirschmann, M., Remus, R.-S., and Teklu, A. F. (2016). Origin and properties of dual and offset active galactic nuclei in a cosmological simulation at z=2. MNRAS, 458(1):1013–1028.

Sunyaev, R. A. and Zeldovich, Y. B. (1970). Small-Scale Fluctuations of Relic Radiation. Ap&SS, 7(1):3–19.

Teklu, A. F., Remus, R.-S., Dolag, K., Beck, A. M., Burkert, A., Schmidt, A. S., Schulze, F., and Steinborn, L. K. (2015). Connecting Angular Momentum and Galactic Dynamics: The Complex Interplay between Spin, Mass, and Morphology. ApJ, 812(1):29.

Teyssier, R. (2002). Cosmological hydrodynamics with adaptive mesh refinement. A new high resolution code called RAMSES. A&A, 385:337–364.

Teyssier, R. (2015). Grid-based hydrodynamics in astrophysical fluid flows. *Annual Review of Astronomy and Astrophysics*, 53:150619171245001.

Thielemann, F. K., Argast, D., Brachwitz, F., Hix, W. R., Höflich, P., Liebendörfer, M., Martinez-Pinedo, G., Mezzacappa, A., Panov, I., and Rauscher, T. (2003). Nuclear cross sections, nuclear structure and stellar nucleosynthesis. Nucl. Phys. A, 718:139–146.

Tian, C. and Chen, Y. (2016). Numerical Simulations of Kelvin-Helmholtz Instability: A Two-dimensional Parametric Study. ApJ, 824(1):60.

Tornatore, L., Borgani, S., Dolag, K., and Matteucci, F. (2007). Chemical enrichment of galaxy clusters from hydrodynamical simulations. MNRAS, 382(3):1050–1072.

Tornatore, L., Borgani, S., Springel, V., Matteucci, F., Menci, N., and Murante, G. (2003). Cooling and heating the intracluster medium in hydrodynamical simulations. MNRAS, 342(4):1025–1040.

Toro, E. (1999). *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Applied mechanics: Researchers and students. Springer.

Toro, E. F., Spruce, M., and Speares, W. (1994). Restoration of the contact surface in the HLL-Riemann solver. *Shock Waves*, 4(1):25–34.

Truelove, J. K., Klein, R. I., McKee, C. F., Holliman, John H., I., Howell, L. H., and Greenough, J. A. (1997). The Jeans Condition: A New Constraint on Spatial Resolution in Simulations of Isothermal Self-gravitational Hydrodynamics. ApJ, 489(2):L179–L183.

Turner, M. S. and Widrow, L. M. (1988). Inflation-produced, large-scale magnetic fields. Phys. Rev. D, 37(10):2743–2754.

Valentini, M., Murante, G., Borgani, S., Granato, G. L., Monaco, P., Brighenti, F., Tornatore, L., Bressan, A., and Lapi, A. (2020). Impact of AGN feedback on galaxies and their multiphase ISM across cosmic time. MNRAS, 491(2):2779–2807.

van den Hoek, L. B. and Groenewegen, M. A. T. (1997). New theoretical yields of intermediate mass stars. A&AS, 123:305–328.

van Leer, B. (1979). Towards the ultimate conservative difference scheme v. a second-order sequel to godunov's method. *Journal of Computational Physics*, 32:101–136.

Vazza, F., Eckert, D., Brüggen, M., and Huber, B. (2015a). Electron and proton acceleration efficiency by merger shocks in galaxy clusters. MNRAS, 451(2):2198–2211.

Vazza, F., Ferrari, C., Brüggen, M., Bonafede, A., Gheller, C., and Wang, P. (2015b). Forecasts for the detection of the magnetised cosmic web from cosmological simulations. A&A, 580:A119.

Viola, M., Monaco, P., Borgani, S., Murante, G., and Tornatore, L. (2008). How does gas cool in dark matter haloes? MNRAS, 383(2):777–790.

Wadsley, J. W., Veeravalli, G., and Couchman, H. M. P. (2008). On the treatment of entropy mixing in numerical cosmology. MNRAS, 387(1):427–438.

Weinberger, R., Springel, V., and Pakmor, R. (2019). The Arepo public code release. *arXiv e-prints*, page arXiv:1909.04667.

Wendland, H. (1995). Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in Computational Mathematics*, 4:389–396.

White, C. J., Stone, J. M., and Gammie, C. F. (2016). An Extension of the Athena++ Code Framework for GRMHD Based on Advanced Riemann Solvers and Staggered-mesh Constrained Transport. ApJS, 225(2):22.

Whitehurst, R. (1995). A free Lagrange method for gas dynamics. MNRAS, 277(2):655–680.

Widrow, L. M. (2002). Origin of galactic and extragalactic magnetic fields. *Reviews of Modern Physics*, 74(3):775–823.

Woosley, S. E. and Weaver, T. A. (1995). The Evolution and Explosion of Massive Stars. II. Explosive Hydrodynamics and Nucleosynthesis. ApJS, 101:181.

Acknowledgements

I would like to express my sincerest gratitude to my first supervisor, PD Dr. Klaus Dolag, for entrusting me with this sophisticated project while always offering a helping hand when I required one. I am deeply humbled by the opportunities I was given, including two scientific exchanges with the university of Trieste, Italy, the participation on his recommendation in a fantastic winter school on Tenerife, Spain, as well as the possibility to continue my scientific career in the CAST group had I chosen so.

I am indebted to Dr. David Hubber for sacrificing countless hours in addition to a demanding job in order to help this project flourish.

I thank my second supervisor, Dr. Rhea-Silvia Remus, for initially inviting me to become a member of the CAST group and for her motivating words throughout the 13 months I spent with them.

Additional thanks I owe to Ludwig Böss for his untiring readiness to answer my questions and for proofreading this work.

I am grateful to Dr. Giuseppe Murante, Luigi Bassini, and Prof. Dr. Stefano Borgani for their hospitality in Trieste as well as countless discussions about the intricacies of meshless schemes.

Lastly, I thank Ulrich Steinwandel for providing me with the ICs for the Rayleigh-Taylor test, Joseph O'Leary for helping me improve my proficiency in `Python`, and Maximillian Kühn for making the office an enjoyable one to work in.

Selbstständigkeitserklärung

Hiermit erkläre ich, die vorliegende Arbeit selbstständig verfasst zu haben und keine anderen als die in der Arbeit angegebenen Quellen und Hilfsmittel benutzt zu haben.

München, 26. April 2020

Paul Marten Hinz