SIMULATION AND VISUALISATION OF ASTROPHYSICAL PLASMAS WITH SMOOTHED PARTICLE MAGNETOHYDRODYNAMICS

TERRENCE S. TRICCO

Simulation and Visualisation of Astrophysical Plasmas with Smoothed Particle Magnetohydrodynamics

by

© Terrence S. Tricco
B.Sc.(Hons)

A thesis submitted to the
School of Graduate Studies
in partial fulfillment of the
requirements for the degree of
Master of Science.

Department of Computational Sciences
Memorial University of Newfoundland

August 11, 2010

St. John's Newfoundland

Contents

Abstract

A numerical implementation of the equations of magnetohydrodynamics has been developed using the Smoothed Particle Hydrodynamics (SPH) method. Self gravitational forces have been included using the Barnes Hut octree algorithm. A nearest neighbour search algorithm has been devised which links adjacent, spatially exclusive nodes from a spatial decomposition tree to create a mesh facilitating efficient searching. Visualisations of SPH fluids were created using Delaunay triangulation with particles serving as triangle vertices.

Acknowledgements

"Two roads diverged in a wood, and I
I took the one less traveled by,
And that has made all the difference."

Robert Frost

I wish to start by expressing my gratitude to my supervisor Dr. John Lewis. His immense patience allowed me to take my Master's research in the direction I desired, and for that I am thankful. I like to believe that I have gained some measure of his wisdom and knowledge during my time with him.

I would also like to thank the other graduate students who have been here during my time. There are many who have been of help to me, in both direct and indirect ways, but I wish to give significant mention to Jason Mercer who was always willing to listen to me talk about whatever I was working on and offer insightful criticisms. As well, it would be remiss not to mention Stephen Hinchey. We've shared parallel journeys through our Master's programs, and having someone to talk to who could relate to your pains and triumphs was always welcome.

Finally, I wish to sincerely thank my mother. Her support through all this time is what has allowed me to reach where I am today.

List of Tables

List of Figures

List of Algorithms

"Well, here we are."

John Boone

1

Introduction

Plasmas are ubiquitous in astrophysics. Almost all visible matter in stars and galaxies is composed of plasma. This state of matter emerges when gasses ionize, that is, when the electrons disassociate from the molecules comprising the gas. Since these charged particles are no longer locked in tandem with each other, magnetic forces are free to play a dominant role in their dynamics. Many of the problems studied in astrophysics are only now being examined with the inclusion of magnetic effects, and the impetus to incorporate their effects into numerical models is great.

For this research, two softwares were developed. The first is an astrophysical magnetohydrodynamic simulation software. This models magnetohydrodynamic fluid mechanics along with self-gravitating forces. The second software consisted of plotting routines and visualisation techniques for the simulated systems.

The primary design focus of the simulation program is to view applied physics as a series of operators acting upon a system of data. This simple idea allows for powerful adaptability in the type of simulation that may be performed. Each simulation run

defines a set of operators that will be used, allowing for choice between which physics to model, and which model to use for those physics. This also provides a systematic avenue for the addition of new physics, allowing the software to grow in a responsible manner.

The code is structured such that the numerical integration scheme holds the set of operators, along with a user chosen adaptive time stepping scheme. The user decides which operators to use, but it is the numerical integration scheme which dictates how those operators are to be used. Similarly for the adaptive time stepping scheme.

The numerical integrator interacts with each operator through a standard set of protocol functions, with the inside workings of these functions being implemented individually per operator. Each operator has an initialisation function, which is run once at the beginning of a simulation, and an operate function which carries the primary workload. Analagously, the numerical integration scheme has an initialisation function, and a step function, which advances a system of data forward one step in time.

The simulation software utilises the Smoothed Particle Hydrodynamic method to model fluid mechanics, and incorporates the latest techniques for simulating magnetohydrodynamics with this method. As will be shown later in this thesis, the Smoothed Particle Hydrodynamic method has many possible implementations, which may have variable component choices, making the operator archetype described previously exceptionally suited for this method. It is currently an active area of interest to improve upon the magnetohydrodynamic implementations of Smoothed Particle Hydrodynamics. The Barnes Hut octree approach for the purposes of calculating gravitational forces has been implemented as an operator choice. Due to the design of the software, it is simple to run a purely hydrodynamic system by excluding magnetic and gravitational operators, or as a pure gravitational N body code by exluding all

hydrodynamic operators.

A necessary step for implementation of Smoothed Particle Hydrodynamics is nearest neighbour searching, and an efficient nearest neighbour search algorithm was developed in this work. The premise of this algorithm is to use the octree generated in the Barnes Hut gravitational routine as an aid for the search. While using spatial decomposition trees have been employed in the past for nearest neighbour searches, the algorithm developed in this thesis uses the tree structure as an intermediary towards creating a search mesh. This search mesh allows for the nearest neighbour search to be conducted with greater efficiency than when using the tree structure directly.

A visualisation technique using a triangulation method was created for the analysis and visualisation software. Using the Delaunay triangulation method, a surface can be generated using the particles of the Smoothed Particle Hydrodynamic method serving as triangle vertices, and values of fluid properties read from the particles can be used as weightings with which to colour the triangles.

1.1 Outline

This thesis is divided into eight chapters. Chapter 2 will discuss plasmas in greater detail. Starting from simple Euler fluid flow and using Maxwell's equations, the equations of magnetohydrodynamics will be built up. Properties of plasmas, and the types of wave motion that are possible in plasmas will be discussed.

Chapter 3 will introduce the numerical fluid model, and show how the equations of magnetohydrodynamics are implemented using it.

Chapter 4 will focus upon the Barnes Hut octree gravitational force algorithm.

Numerical methods which are not a core part of the previous numerical models will be discussed in chapter 5. These include the embedded scripting used for simulation

initialisation, the nearest neighbour search algorithm, numerical integration methods, and the approaches for adaptive time stepping.

Chapter 6 will discuss the visualisation techniques used for Smoothed Particle Hydrodynamics.

Chapter 7 will present the test cases which were used development of the simulation program to verify its correctness.

The final chapter, Chapter 8, will conclude the thesis. Considerations towards future work will be examined.

"Who can compare with him in kingliness?
Who can say like Gilgamesh: I am King! ?"

The Epic of Gilgamesh

2

Magnetohydrodynamics

Magnetohydrodynamics (MHD) is the merger of hydrodynamics with electrodynamic theory, and describes the flow of conducting fluids. This treats the fluid macroscopically, where the dynamics at the molecular level are considered unimportant. Other approaches are possible, notably various types of kinetic theory, in which the fluid system is treated in a statistical manner, but since we are interested in studying astrophysical plasmas, the length scales considered are large enough, and frequencies small enough, to justify using a fluid model.

MHD can be formulated as either a two fluid model, in which the plasma is described by two mixed fluids, one consisting of the negatively charged particles and the second the positively charged particles, or as a single fluid model of the bulk material. For many astrophysical applications, the distribution of negative and positive charges are effectively homogeneous for the length scales of interest, and the single fluid model is sufficient.

The MHD description used in this thesis is that of ideal MHD, for which the con-

ductivity of the plasma is considered infinite. This is an acceptable approximation for many plasmas. We begin by discussing the Euler fluid, into which we will integrate electromagnetism.

2.1 Euler Flow

The simplest fluid description is that given by Euler in 1755, and is known as Euler flow. It regards the fluid as a continuous medium, which has the property that any arbitrary section of the medium is itself a continuum. While all fluids are made of molecules, and are anything but continuous at this resolution, we suppose that any small element of the fluid under examination is considered large in relation to inter-molecular distances. This allows for differential calculus to be a valid method of analysis of the fluid.

Euler flow is ideal, in that the system contains no dissipation, either through the viscosity of the fluid or through thermal conduction. This means its motion must be adiabatic, as there are no sources of entropy generation.

We now discuss the equations governing the rate of change of density and velocity of an element of fluid, which combined with an equation of state, form a complete system of equations. The presentation of these equations follows that of Landau and Lifshitz [25].

2.1.1 Continuity Equation

Consider a volume of fluid V_0 enclosed by a surface S. The total mass flowing out of the volume can be given by

$$\oint_S \rho \mathbf{v} \cdot d\mathbf{s} \tag{2.1}$$

where the integration is over the surface S. This decrease in the mass of the volume may also be written as

$$-\frac{\partial}{\partial t} \int \rho dV, \tag{2.2}$$

and by equating the two we have

$$-\frac{\partial}{\partial t} \int \rho dV = \oint \rho \mathbf{v} \cdot d\mathbf{s}. \tag{2.3}$$

Using Green's theorem, we can transform the surface integral to a volume integral, yielding

$$-\frac{\partial}{\partial t} \int \rho dV = \int \boldsymbol{\nabla} \cdot (\rho \mathbf{v})\, dV \tag{2.4}$$

and therefore

$$\int \left(\frac{\partial}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \mathbf{v}) \right) dV = 0. \tag{2.5}$$

Since equation 2.5 is true for any arbitrary volume, this implies that

$$\frac{\partial \rho}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \mathbf{v}) = 0. \tag{2.6}$$

Equation 2.6 is known as the equation of continuity, and states that the mass in any portion of fluid can only change by mass flowing into or out of that portion. Taken for the system as a whole, this implies mass is conserved.

Expanding the second term, and introducing the material derivative

$$\left(\frac{d}{dt}\right) = \left(\frac{\partial}{\partial t} + \mathbf{v} \cdot \boldsymbol{\nabla}\right), \tag{2.7}$$

we can write the continuity equation as

$$\frac{d\rho}{dt} = -\rho \boldsymbol{\nabla} \cdot \mathbf{v}. \tag{2.8}$$

The material derivative denotes the rate of change of a quantity, ρ in this case, for an element of fluid along a streamline.

2.1.2 Momentum Equation

As with the continuity equation, we begin by considering a volume of fluid V_0 enclosed by a surface S. The total force exerted by thermal pressure P on this surface is given by

$$-\oint P d\mathbf{s} \tag{2.9}$$

which can be converted to a volume integral to yield

$$-\oint P d\mathbf{s} = -\int \boldsymbol{\nabla} P dV. \tag{2.10}$$

From this, we can state that the force exerted on any element of fluid is $-\boldsymbol{\nabla} P$. By Newton's law, this means then that

$$\rho \frac{d\mathbf{v}}{dt} = -\boldsymbol{\nabla} P, \tag{2.11}$$

where d/dt is the material derivative, and equation 2.11 is known as the momentum equation.

2.2 Maxwell's Equations

Maxwell's equations are four of the fundamental equations of classical electrodynamic theory. In SI units, they are

$$\nabla \cdot \mathbf{E} = \frac{\tau}{\epsilon_0} \tag{2.12}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{2.13}$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \tag{2.14}$$

$$\nabla \times \mathbf{B} = \mu_0 \mathbf{J} + \mu_0 \epsilon_0 \frac{\partial \mathbf{E}}{\partial t} \tag{2.15}$$

where are referred to, in order, as Gauss's law, the div $\boldsymbol{B}$ constraint[1], Faraday's law, and Ampère's law. These equations describes the properties and evolutions of the electric field, $\mathbf{E}$, and magnetic field, $\mathbf{B}$, in terms of their sources, charge density τ and current density $\mathbf{J}$. The parameters ϵ_0 and μ_0 are the permitivity and permeability of free space.

2.3 Ideal Magnetohydrodynamics

Ideal MHD is derived under the assumption that the conductivity of the fluid is infinite. This is the first major assumption imposed. The second major assumption is that the distribution of negative and positive charges are effectively homogeneous. While some fluctuations in the charge density are sure to exist, the time and length scales are large enough to average these out. Under these circumstances, the charge density

[1] In literature on electromagnetics, this is usually either referred to as "the absence of free magnetic monopoles", or "Gauss's law for magnetism", of which both are equally apt. However, in computational settings, it becomes convienient to refer to it as above.

can be neglected, and without a source, the electric field is taken to be negligable. This is in agreement with the electric field inside a perfect conductor.

Where not otherwise specified, several textbook sources [10, 18, 5, 7] have served as references for the following material.

2.3.1 Momentum Equation

The motion of plasma is governed by forces generated from the thermal pressure, as well as forces arising from the magnetic fields. We will derive the form for the latter forces starting from the Lorentz force law.

Given as

$$\mathbf{F} = q\left(\mathbf{E} + \mathbf{v} \times \mathbf{B}\right), \tag{2.16}$$

the Lorentz force law describes the force exerted on a particle of charge q moving at velocity $\mathbf{v}$ by the electric and magnetic fields. Under the assumptions of ideal MHD, the electric field contribution to the force can be taken to be zero. We recast equation 2.16 as

$$\mathbf{f} = \mathbf{J} \times \mathbf{B}, \tag{2.17}$$

which is given in terms of force per unit volume.

Substituting Ampère's law into equation 2.17, where $\partial \mathbf{E}/\partial t = 0$, we obtain

$$\mathbf{f} = \frac{1}{\mu_0}\left(\boldsymbol{\nabla} \times \mathbf{B}\right) \times \mathbf{B}. \tag{2.18}$$

Using the vector calculus identity

$$(\boldsymbol{\nabla} \times \mathbf{A}) \times \mathbf{A} = (\mathbf{A} \cdot \boldsymbol{\nabla})\,\mathbf{A} - \tfrac{1}{2}\boldsymbol{\nabla}\,(\mathbf{A} \cdot \mathbf{A})\,, \tag{2.19}$$

we can rewrite equation 2.18 as

$$\mathbf{f} = \frac{1}{\mu_0}\left((\mathbf{B} \cdot \boldsymbol{\nabla})\,\mathbf{B} - \frac{1}{2}\boldsymbol{\nabla} B^2\right). \tag{2.20}$$

Including a $(\boldsymbol{\nabla} \cdot \mathbf{B})\mathbf{B}$ term, which equals zero via the div $\boldsymbol{B}$ constraint, it becomes possible to express equation 2.20 in tensor notation. Declaring the Maxwell stress tensor to be

$$M^{ij} = \frac{1}{\mu_0}\left(B^i B^j - \frac{1}{2}\delta_{ij} B^2\right), \tag{2.21}$$

we can write equation 2.20 as

$$\mathbf{f} = \boldsymbol{\nabla} \cdot \mathbf{M}. \tag{2.22}$$

Formulating the Lorentz force law in this way is attractive because it describes the force entirely in terms of the magnetic field. The underlying currents which are generating those fields do not need to be considered. With this, it is easy to include the Lorentz force in the hydrodynamic model yielding the momentum equation

$$\frac{d\mathbf{v}}{dt} = -\frac{1}{\rho}\boldsymbol{\nabla} P + \frac{1}{\rho}\boldsymbol{\nabla} \cdot \mathbf{M}. \tag{2.23}$$

2.3.2 Magnetic Pressure

If we define a local coordinate system such that the magnetic field is directed along a given axis, say $\mathbf{B} = B\hat{\mathbf{z}}$, the Maxwell stress tensor simplifies to

$$\mathbf{M} = \begin{pmatrix} -B^2/(2\mu_0) & 0 & 0 \\ 0 & -B^2/(2\mu_0) & 0 \\ 0 & 0 & +B^2/(2\mu_0) \end{pmatrix}. \tag{2.24}$$

From this, we can see that there exists a repulsive force of $-B^2/(2\mu_0)$ in all directions, which acts like a pressure term and is accordingly called the magnetic pressure. However, a second force, $B^2/(\mu_0)$, exists which is directed along the magnetic field lines. This force is analagous to a tension exerted on a string, and is referred to as the magnetic tension.

2.3.3 Induction Equation

To describe the evolution of the magnetic field inside the plasma, we can use Faraday's law. The difficulty however, is that this describes the time change of the magnetic field in terms of the electric field. Remember that the assumption about the electric field is not that it is precisely zero, just that it is weak enough to have its effect be considered negligable in comparison to effects from the magnetic field.

To remove mention of the electric field, we can use Ohm's law which describes the current density as

$$\mathbf{J} = \sigma \left(\mathbf{E} + \mathbf{v} \times \mathbf{B} \right). \tag{2.25}$$

As the conductivity in ideal MHD is taken to be infinite, yet $\mathbf{J}$ remains finite, this implies that $(\mathbf{E} + \mathbf{v} \times \mathbf{B}) \to 0$. This allows us to replace the electric field in Faraday's equation with $(-\mathbf{v} \times \mathbf{B})$, yielding

$$\frac{\partial \mathbf{B}}{\partial t} = \boldsymbol{\nabla} \times \left(\mathbf{v} \times \mathbf{B} \right), \tag{2.26}$$

and is known as the induction equation.

2.3.4 Frozen Field Theorem

An interesting consequence exists because of the induction equation. Considering a surface $S(t)$ bounded by a closed contour $C(t)$ which moves with the fluid, the rate of change of magnetic flux through the surface is given by

$$\frac{d}{dt}\int_{S(t)} \mathbf{B}(\mathbf{r},t)\cdot d\mathbf{s}. \tag{2.27}$$

We can use Leibniz's rule in two dimensions to bring the differentiation inside the integral, obtaining

$$\frac{d}{dt}\int_{S(t)} \mathbf{B}(\mathbf{r},t)\cdot d\mathbf{s} = \int_{S(t)} \frac{\partial \mathbf{B}}{\partial t}\cdot d\mathbf{s} + \oint_{C(t)} \mathbf{B}\cdot(\mathbf{v}_C\times d\mathbf{l}), \tag{2.28}$$

where $\mathbf{v}_C$ is the velocity of the contour. Using the vector identity $\mathbf{A}\cdot(\mathbf{B}\times\mathbf{C}) = \mathbf{C}\cdot(\mathbf{A}\times\mathbf{B})$, and subsequently applying Stoke's theorem, we can rewrite the line integral in equation 2.28 to a surface integral, yielding

$$\begin{aligned}
\frac{d}{dt}\int_{S(t)} \mathbf{B}(\mathbf{r},t)\cdot d\mathbf{s} &= \int_{S(t)} \frac{\partial \mathbf{B}}{\partial t}\cdot d\mathbf{s} + \oint_{C(t)} (\mathbf{B}\times\mathbf{v}_C)\cdot d\mathbf{l} \\
&= \int_{S(t)} \frac{\partial \mathbf{B}}{\partial t}\cdot d\mathbf{s} + \int_{S(t)} \boldsymbol{\nabla}\times(\mathbf{B}\times\mathbf{v}_C)\cdot d\mathbf{s} \\
&= \int_{S(t)} \frac{\partial \mathbf{B}}{\partial t}\cdot d\mathbf{s} - \int_{S(t)} \boldsymbol{\nabla}\times(\mathbf{v}_C\times\mathbf{B})\cdot d\mathbf{s} \\
&= \int_{S(t)} \left(\frac{\partial \mathbf{B}}{\partial t} - \boldsymbol{\nabla}\times(\mathbf{v}_C\times\mathbf{B})\right)\cdot d\mathbf{s}.
\end{aligned} \tag{2.29}$$

However, by the induction equation, this then equals zero. Thus, the magnetic flux through any arbitrary portion of the fluid remains constant in time. Owing to

this, it is said that the magnetic field lines are frozen in to the fluid, and are dragged along with the motion of the fluid.

2.3.5 Complete Set of Ideal MHD Equations

We now present a summary of the set of ideal MHD equations:

$$\frac{d\rho}{dt} = -\rho \nabla \cdot \mathbf{v}, \tag{2.30}$$

$$\frac{d\mathbf{v}}{dt} = -\frac{1}{\rho}\nabla P + \frac{1}{\rho}\nabla \cdot \mathbf{M}, \tag{2.31}$$

$$\frac{\partial \mathbf{B}}{\partial t} = \nabla \times (\mathbf{v} \times \mathbf{B}), \tag{2.32}$$

$$\nabla \cdot \mathbf{B} = 0, \tag{2.33}$$

$$P = f(\rho). \tag{2.34}$$

These describe the rate of change for the density, velocity, and magnetic field within the plasma through the continuity, momentum, and induction equations. The div $\boldsymbol{B}$ constraint is present, which enforces restrictions on the magnetic field. Lastly, an equation of state is used to close the set of equations.

2.4 Magnetohydrodynamic Waves

In hydrodynamic systems, the only type of wave propagation that exists is of longitudinal sound waves. However the addition of magnetic forces allow for a variety of other wave types.

To demonstrate these waves, we consider a system at equilibrium in which small perturbations are added to the density, magnetic field, and velocity of the form

$$\rho(\mathbf{r}, t) = \rho_0 + \rho_1(\mathbf{r}, t), \tag{2.35}$$

$$\mathbf{B}(\mathbf{r}, t) = \mathbf{B}_0 + \mathbf{B}_1(\mathbf{r}, t), \tag{2.36}$$

$$\mathbf{v}(\mathbf{r}, t) = \mathbf{v}_1(\mathbf{r}, t). \tag{2.37}$$

The equilibrium values $\mathbf{B}_0$ and ρ_0 are uniform and constant, and the fluid medium is otherwise at rest.

Inserting equations 2.35, 2.36, and 2.37 into the MHD equations, and neglecting terms of second order, we obtain

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \mathbf{v} \cdot \boldsymbol{\nabla} \rho &= -\rho \boldsymbol{\nabla} \cdot \mathbf{v} \\ \frac{\partial \rho_0}{\partial t} + \frac{\partial \rho_1}{\partial t} &= -\rho_0 \boldsymbol{\nabla} \cdot \mathbf{v}_1 - \rho_1 \boldsymbol{\nabla} \cdot \mathbf{v}_1 - \mathbf{v}_1 \boldsymbol{\nabla} \rho_0 - \mathbf{v}_1 \boldsymbol{\nabla} \rho_1 \\ \frac{\partial \rho_1}{\partial t} &= -\rho_0 \boldsymbol{\nabla} \cdot \mathbf{v}_1, \end{aligned} \tag{2.38}$$

$$\begin{aligned} \frac{\partial \mathbf{B}}{\partial t} &= \boldsymbol{\nabla} \times \mathbf{v} \times \mathbf{B} \\ \frac{\partial \mathbf{B}_0}{\partial t} + \frac{\partial \mathbf{B}_1}{\partial t} &= \boldsymbol{\nabla} \times \mathbf{v}_1 \times (\mathbf{B}_0 + \mathbf{B}_1) \\ \frac{\partial \mathbf{B}_1}{\partial t} &= \boldsymbol{\nabla} \times \mathbf{v}_1 \times \mathbf{B}_0, \end{aligned} \tag{2.39}$$

and, using $\boldsymbol{\nabla} P = c^2 \boldsymbol{\nabla} \rho$,

$$\rho\frac{\partial \mathbf{v}}{\partial t} + \rho\left(\mathbf{v}\cdot\boldsymbol{\nabla}\right)\mathbf{v} = -\boldsymbol{\nabla}P + \frac{1}{\mu_0}\left(\boldsymbol{\nabla}\times\mathbf{B}\right)\times\mathbf{B}$$

$$\begin{aligned}
\left(\rho_0+\rho_1\right)\frac{\partial \mathbf{v}_1}{\partial t} + \left(\rho_0+\rho_1\right)\left(\mathbf{v}_1\cdot\boldsymbol{\nabla}\right)\mathbf{v}_1 &= -c^2\boldsymbol{\nabla}\left(\rho_0+\rho_1\right) \\
&\quad + \frac{1}{\mu_0}\left(\boldsymbol{\nabla}\times\left(\mathbf{B}_0+\mathbf{B}_1\right)\right)\times\left(\mathbf{B}_0+\mathbf{B}_1\right) \\
\rho_0\frac{\partial \mathbf{v}_1}{\partial t} &= -c^2\boldsymbol{\nabla}\rho_1 + \frac{1}{\mu_0}\left(\boldsymbol{\nabla}\times\mathbf{B}_1\right)\times\left(\mathbf{B}_0+\mathbf{B}_1\right) \\
\frac{\partial \mathbf{v}_1}{\partial t} &= -\frac{c^2}{\rho_0}\boldsymbol{\nabla}\rho_1 - \frac{1}{\mu_0\rho_0}\mathbf{B}_0\times\left(\boldsymbol{\nabla}\times\mathbf{B}_1\right). \qquad (2.40)
\end{aligned}$$

Taking the time derivative of 2.40,

$$\frac{\partial^2 \mathbf{v}_1}{\partial t^2} = -\frac{c^2}{\rho_0}\boldsymbol{\nabla}\frac{\partial \rho_1}{\partial t} - \frac{1}{\mu_0\rho_0}\mathbf{B}_0\times\left(\boldsymbol{\nabla}\times\frac{\partial \mathbf{B}_1}{\partial t}\right), \qquad (2.41)$$

equations 2.38 and 2.39 can be substituted to yield

$$\frac{\partial^2 \mathbf{v}_1}{\partial t^2} - c^2\boldsymbol{\nabla}(\boldsymbol{\nabla}\cdot\mathbf{v}_1) + \mathbf{v}_a\times\left(\boldsymbol{\nabla}\times\left(\boldsymbol{\nabla}\times\left(\mathbf{v}_1\times\mathbf{v}_a\right)\right)\right) = 0. \qquad (2.42)$$

Here we have introduced the Alfvén velocity, given as

$$\mathbf{v}_a = \frac{\mathbf{B}_0}{\sqrt{\mu_0\rho_0}}. \qquad (2.43)$$

Equation 2.42 can be seen to have plane wave solutions of the form

$$\mathbf{v}_1 = \mathbf{v}_k \exp\left(i\mathbf{k}\cdot\mathbf{r} - i\omega t\right). \qquad (2.44)$$

Inserting 2.44 into 2.42, we see that the $\partial/\partial t$ operators are replaced by $(-i\omega)$, and $\boldsymbol{\nabla}$ by $(i\mathbf{k})$, which yields

$$\omega^2 \mathbf{v}_k - c^2 (\mathbf{k} \cdot \mathbf{v}_k) \mathbf{k} + \mathbf{v}_a \times (\mathbf{k} \times (\mathbf{k} \times (\mathbf{v}_k \times \mathbf{v}_a))) = 0. \tag{2.45}$$

Using the vector identity $A \times (B \times C) = (A \cdot C) B - (A \cdot B) C$, we can rewrite the third term as

$$\begin{aligned}
&= \mathbf{v}_a \times (\mathbf{k} \times (\mathbf{k} \times (\mathbf{v}_k \times \mathbf{v}_a))) \\
&= \mathbf{v}_a \times (\mathbf{k} \times [(\mathbf{k} \cdot \mathbf{v}_a) \mathbf{v}_k - (\mathbf{k} \cdot \mathbf{v}_k) \mathbf{v}_a]) \\
&= (\mathbf{k} \cdot \mathbf{v}_a) \mathbf{v}_a \times (\mathbf{k} \times \mathbf{v}_k) - (\mathbf{k} \cdot \mathbf{v}_k) \mathbf{v}_a \times (\mathbf{k} \times \mathbf{v}_a) \\
&= (\mathbf{k} \cdot \mathbf{v}_a) [(\mathbf{v}_a \cdot \mathbf{v}_k) \mathbf{k} - (\mathbf{v}_a \cdot \mathbf{k}) \mathbf{v}_k] - (\mathbf{k} \cdot \mathbf{v}_k) [(\mathbf{v}_a \cdot \mathbf{v}_a) \mathbf{k} - (\mathbf{v}_a \cdot \mathbf{k}) \mathbf{v}_a] \\
&= (\mathbf{k} \cdot \mathbf{v}_a) (\mathbf{v}_a \cdot \mathbf{v}_k) \mathbf{k} - (\mathbf{k} \cdot \mathbf{v}_a) (\mathbf{v}_a \cdot \mathbf{k}) \mathbf{v}_k \\
&\quad - (\mathbf{k} \cdot \mathbf{v}_k) (\mathbf{v}_a \cdot \mathbf{v}_a) \mathbf{k} + (\mathbf{k} \cdot \mathbf{v}_k) (\mathbf{v}_a \cdot \mathbf{k}) \mathbf{v}_a.
\end{aligned} \tag{2.46}$$

Inserting this into equation 2.45, and collecting the like terms of $\mathbf{v}_k$, $\mathbf{v}_a$, and $\mathbf{k}$, we obtain

$$\begin{aligned}
&\left[\omega^2 - (\mathbf{v}_a \cdot \mathbf{k})^2\right] \mathbf{v}_k + \left[(\mathbf{k} \cdot \mathbf{v}_k) (\mathbf{k} \cdot \mathbf{v}_a) \right] \mathbf{v}_a \\
&\quad - \left[\left(c^2 + v_a^2\right) (\mathbf{k} \cdot \mathbf{v}_k) - (\mathbf{k} \cdot \mathbf{v}_a) (\mathbf{v}_a \cdot \mathbf{v}_k) \right] \mathbf{k} = 0.
\end{aligned} \tag{2.47}$$

Equation 2.47 is a set of three equations for the three components of the wave vector $\mathbf{v}_k$. Also, note that if $\mathbf{B}_0 = 0$, this reduces to

$$\omega^2 \mathbf{v}_k - c^2 (\mathbf{k} \cdot \mathbf{v}_1) \mathbf{k} = 0 \tag{2.48}$$

and ordinary sound waves are produced with phase velocity $\omega/k = c$.

Following the example of Boyd and Sanderson [10], we can choose the coordinate system such that, without any loss of generality, the z axis is along the direction of the magnetic field lines, with the direction of wave propagation selected to lie in the y-z plane. This allows the wave vector to be parallel, orthogonal, or of some intermediate angle to the magnetic field. Succinctly stated,

$$\mathbf{v}_a = v_a \hat{\mathbf{z}}, \tag{2.49}$$

$$\mathbf{k} = k_\perp \hat{\mathbf{y}} + k_\parallel \hat{\mathbf{z}}. \tag{2.50}$$

With these definitions, the $\mathbf{v}_k$, $\mathbf{k}$, and $\mathbf{v}_a$ components of 2.47 reduce to

$$\begin{aligned} &= \left[\omega^2 - \left(v_a \hat{\mathbf{z}} \cdot \left(k_\perp \hat{\mathbf{y}} + k_\parallel \hat{\mathbf{z}}\right)\right)^2\right] \mathbf{v}_k \\ &= \left[\omega^2 - v_a^2 k_\parallel^2\right] \left(v_x \hat{\mathbf{x}} + v_y \hat{\mathbf{y}} + v_z \hat{\mathbf{z}}\right), \end{aligned} \tag{2.51}$$

$$\begin{aligned} &= -\Big[\left(c^2 + v_a^2\right) \left(k_\perp \hat{\mathbf{y}} + k_\parallel \hat{\mathbf{z}}\right) \cdot \left(v_x \hat{\mathbf{x}} + v_y \hat{\mathbf{y}} + v_z \hat{\mathbf{z}}\right) \\ &\qquad - \left(k_\perp \hat{\mathbf{y}} + k_\parallel \hat{\mathbf{z}}\right) \cdot \left(v_x \hat{\mathbf{x}} + v_y \hat{\mathbf{y}} + v_z \hat{\mathbf{z}}\right) \Big] \mathbf{k} \\ &= -\Big[\left(c^2 + v_a^2\right) \left(k_\perp v_y + k_\parallel v_z\right) - v_a^2 k_\parallel v_z \Big] \left(k_\perp \hat{\mathbf{y}} + k_\parallel \hat{\mathbf{z}}\right) \\ &= -\Big[\left(c^2 + v_a^2\right) k_\perp v_y + c^2 k_\parallel v_z \Big] \left(k_\perp \hat{\mathbf{y}} + k_\parallel \hat{\mathbf{z}}\right), \end{aligned} \tag{2.52}$$

and

$$
\begin{aligned}
&= \Big[\left(k_\perp \hat{\mathbf{y}} + k_\parallel \hat{\mathbf{z}}\right) \cdot \left(v_x \hat{\mathbf{x}} + v_y \hat{\mathbf{y}} + v_z \hat{\mathbf{z}}\right) \left(k_\perp \hat{\mathbf{y}} + k_\parallel \hat{\mathbf{z}}\right) \cdot \left(v_a \hat{\mathbf{z}}\right) \Big] \mathbf{v}_a \\
&= \Big[\left(k_\perp v_y + k_\parallel v_a\right) \left(k_\parallel v_a\right) \Big] v_a \hat{\mathbf{z}} \\
&= \Big[v_a k_\perp k_\parallel v_y + v_a k_\parallel^2 v_z \Big] v_a \hat{\mathbf{z}}. \qquad (2.53)
\end{aligned}
$$

This leads to the system of equations

$$
\begin{pmatrix}
\omega^2 - v_a^2 k_\parallel^2 & 0 & 0 \\
0 & \omega^2 - c^2 k_\perp^2 - v_a^2 k^2 & -c^2 k_\parallel k_\perp \\
0 & -c^2 k_\parallel k_\perp & \omega^2 - c^2 k_\parallel^2
\end{pmatrix}
\begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix}
=
\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \qquad (2.54)
$$

To have non-trivial solutions, the determinant of the coefficients must equal zero. This yields the dispersion relation

$$
\left(\omega^2 - v_a^2 k_\parallel^2\right) \left(\omega^4 - \omega^2 k^2 \left(c^2 + v_a^2\right) + v_a^2 c^2 k^2 k_\parallel^2\right) = 0. \qquad (2.55)
$$

2.4.1 Pure Alvén Waves

The first mode, $\left(\omega^2 - v_a^2 k_\parallel^2\right)$, is decoupled from the others, and it oscillates along the x axis, orthogonal to both the magnetic field and the direction of wave propagation. Therefore, this is a transverse wave.

Recognizing that the $v_a^2 k_\parallel^2$ term was originally $\left(\mathbf{v}_a \cdot \mathbf{k}\right)^2$, we can instead write the dot product as $\mathbf{v}_a \cdot \mathbf{k} = |v_a|\,|k| \cos\theta$ to obtain the phase velocity

$$
\frac{\omega}{k} = v_a \cos\theta, \qquad (2.56)
$$

where θ is the angle between the magnetic field and the wave vector. When these are

parallel to each other, the phase velocity is at a maximum of v_a, the Alfvén speed, and vanishes when perpendicular.

Previously we noted that there exists a tension force along the magnetic field lines of B^2/μ_0. By analogy with elastic strings, it is unsurprising that there should exist transverse waves which propagate along the magnetic field lines. The transverse wave motion of elastic strings has phase velocity $\omega/k = \sqrt{T/\rho}$ where T is the tension on the string. Replacing T by the magnetic tension, we obtain the Alfvén velocity.

2.4.2 Magnetoacoustic Waves

The other modes can be solved to obtain phase velocities of

$$\frac{\omega^2}{k^2} = \frac{1}{2}\left(c^2 + v_a^2\right) \pm \frac{1}{2}\left[\left(c^2 + v_a^2\right)^2 - 4c^2 v_a^2 \frac{k_\parallel^2}{k^2}\right]^{1/2}. \tag{2.57}$$

These two modes, called the fast and slow Alfvén waves, propagate along the direction of the wave vector and are longitudinal. Due to this, and because of the coupling of the speed of sound and the Alfvén velocity, they are referred to as magnetoacoustic or magnetosonic waves.

When the magnetic field is perpendicular to the wave vector, that is $\mathbf{k} = k_\perp \hat{\mathbf{y}}$, equation 2.57 simplifies to

$$\frac{\omega^2}{k^2} = \frac{1}{2}\left(c^2 + v_a^2\right) \pm \frac{1}{2}\left(c^2 + v_a^2\right). \tag{2.58}$$

Thus the fast mode will have velocity $\sqrt{c^2 + v_a^2}$, while the slow mode vanishes for this case.

The other extreme is when the magnetic field and wave vector are parallel to each other. For this case, $\mathbf{k} = k_\parallel \hat{\mathbf{z}}$, and equation 2.57 will become

$$\frac{\omega^2}{k^2} = \frac{1}{2}\left(c^2 + v_a^2\right) \pm \frac{1}{2}\left[c^4 + v_a^4 - 2c^2 v_a^2\right]^{1/2}. \tag{2.59}$$

This presents an interesting scenario, in that the square root term can be decomposed in two ways: as $(c^2 - v_a^2)$, or as $(v_a^2 - c^2)$. Either way, this yields the two solutions

$$\frac{\omega}{k} = c, \quad \frac{\omega}{k} = v_a,$$

which are pure sound waves and pure Alfén waves. However, it is not possible to definitively label either of these as the fast or slow MHD waves. Instead, the usual practice is to refer to v_a as the fast wave if $v_a > c$, or c as the fast wave if $c > v_a$, and vice versa for the slow wave.

From these two cases, we can see that the fast wave is at a maximum of $\sqrt{c^2 + v_a^2}$ when the magnetic field is parallel to the wave propagation, decreasing to v_a (or c, if $c > v_a$) when they are perpendicular. Conversely, the slow wave is at a minimum of 0 when they are parallel, increasing to c (or v_a, if $c > v_a$) when they are perpendicular.

"For this reason, the deliberations of the wise commander are sure to assess jointly both advantages and disadvantages. In taking full acount of what is advantageous, he can fulfill his responsibilities; in taking full account of what is disadvantageous, his difficulties become resolvable."

Sun Tzu

3

Smoothed Particle Hydrodynamics

Smoothed Particle Hydrodynamics (SPH) is a Lagrangian method for solving the partial differential equations of hydrodynamics developed by Gingold and Monaghan [17], and independently by Lucy [27] in 1979. It models a fluid by discretizing it into a set of particles containing the mass, momentum, energy, and other properties of the fluid. Hydrodynamical forces act upon the particles, moving them such that they naturally model fluid flow. In these respects, SPH is similar to molecular dynamics simulation, and indeed many of the approaches used in molecular dynamics can be adapted to work with SPH.

The key difference between SPH and molecular dynamics simulations is that the SPH particles have some spatial extent over which their properties are smoothed, giving them a type of "fuzziness". This volume encompasses a number of neighbouring particles, which all combined form an overlapping patchwork of smoothing volumes. It is precisely this which bridges the gap between continuum and fragmentation, allowing this particle based method to function as a quasi-continuum.

3.1 Interpolation

As each particle is smoothed over its local volume, a fluid property, for example $A(\mathbf{r})$, at any point in our model fluid will have a value which is blended or smoothed from particles contributing at this point. For a continuum fluid, the equation

$$A(\mathbf{r}) = \int A(\mathbf{r}')\delta(\mathbf{r} - \mathbf{r}')d\mathbf{r}', \tag{3.1}$$

is used to obtain the value of A at $\mathbf{r}$, where the Dirac delta function acts to "pull out" the value at $\mathbf{r}$. In SPH, the Dirac delta function is replaced with an interpolating kernel $W(\mathbf{r}, h)$ to obtain an integral interpolant

$$A_i(\mathbf{r}) = \int A(\mathbf{r}')W(\mathbf{r} - \mathbf{r}', h)d\mathbf{r}' \tag{3.2}$$

where h is the radius of the smoothing volume, known as the smoothing length.

The interpolating kernel has properties

$$\int W(\mathbf{r} - \mathbf{r}', h)d\mathbf{r}' = 1 \tag{3.3}$$

and

$$\lim_{h\to 0} W(\mathbf{r} - \mathbf{r}', h) = \delta(\mathbf{r} - \mathbf{r}'). \tag{3.4}$$

Since the system is modeled by a set of discrete particles, the integral interpolant is replaced by a summation over the particles

$$A_a(\mathbf{r}_a) = \sum_b m_b \frac{A_b}{\rho_b} W(\mathbf{r}_a - \mathbf{r}_b, h). \tag{3.5}$$

Here, the quantity m_b/ρ_b, that is the mass and density of particle b, acts as the

volume element of the integral. This summation interpolation is the basis for all SPH formulations.

Taking the derivative of $A_a(\mathbf{r})$, we see that it is only applied to the smoothing kernel,

$$\frac{\partial A_a}{\partial x} = \sum_b m_b \frac{A_b}{\rho_b} \frac{\partial W(\mathbf{r}_a - \mathbf{r}_b, h)}{\partial x}. \tag{3.6}$$

Since $W(\mathbf{r}_{ab}, h)$ is an analytic function, the derivative of $A(\mathbf{r})$ is exact. Note that the shorthand notation $\mathbf{r}_{ab}$ has been used to denote the difference $\mathbf{r}_a - \mathbf{r}_b$.

However, if $A(\mathbf{r})$ is a constant function, while the derivative should equal zero, the previous equation will be non-zero. This can be circumvented by using the identity

$$\frac{\partial A}{\partial x} = \frac{1}{\phi}\left(\frac{\partial(\phi A)}{\partial x} - A\frac{\partial \phi}{\partial x}\right), \tag{3.7}$$

where ϕ is any differentiable function. In SPH form, the above becomes

$$\frac{\partial A}{\partial x} = \frac{1}{\phi_a}\sum_b \frac{m_b}{\rho_b}\phi_b \left(A_b - A_a\right) \frac{\partial W(\mathbf{r}_{ab}, h)}{\partial x} \tag{3.8}$$

which vanishes for constant A.

For function ϕ equal to 1, this yields

$$\frac{\partial A}{\partial x} = \sum_b \frac{m_b}{\rho_b}\left(A_b - A_a\right) \frac{\partial W(\mathbf{r}_{ab}, h)}{\partial x}. \tag{3.9}$$

Alternatively, for $\phi = \rho$,

$$\frac{\partial A}{\partial x} = \frac{1}{\rho_a}\sum_b m_b \left(A_b - A_a\right) \frac{\partial W(\mathbf{r}_{ab}, h)}{\partial x}. \tag{3.10}$$

These are both equally valid forms of the derivative definition, just as equation 3.6

was above. The important aspect is that equation 3.9 has much nicer properties than equation 3.6. This will be the first example of how different SPH formulisms can be derived from the same governing equations.

3.1.1 M4 Kernel

Early work in SPH used Gaussian functions for the smoothing kernel. Today, the M4 cubic spline introduced by Monaghan and Lattanzio [31] has become the most widely used. This function, given by

$$W(r,h) = \frac{\sigma}{h^v}\begin{cases} 1 - 6\left(\frac{r}{h}\right)^2 + 6\left(\frac{r}{h}\right)^3 & 0 \leq \frac{r}{h} \leq 0.5 \\ 2\left(1 - \frac{r}{h}\right)^3 & 0.5 < \frac{r}{h} \leq 1 \\ 0 & \frac{r}{h} > 1 \end{cases} \tag{3.11}$$

is piecewise continuous for both the first and second derivatives, and goes to zero at the smoothing length. The parameter v is the number of the dimensions of the system, and σ is the normalisation with values $2/3$, $10/(7\pi)$, and $1/\pi$ for one, two, and three dimensions respectively.

This kernel is radial, such that it only depends upon the absolute value of $\mathbf{r}_a - \mathbf{r}_b$ and $W(\mathbf{r}_{ab}, h) = W(\mathbf{r}_{ba}, h)$. We also note that the spatial derivative of the kernel may be written as

$$\frac{\partial W(\mathbf{r}_{ab}, h)}{\partial \mathbf{r}_a} = \mathbf{r}_{ab} F_{ab}, \tag{3.12}$$

where $F_{ab} \leq 0$, and is a scalar function of $|\mathbf{r}_{ab}|$. By interchanging the indices a and b, this allows for clear understanding that

$$\frac{\partial W(\mathbf{r}_{ab}, h)}{\partial \mathbf{r}_a} = -\frac{\partial W(\mathbf{r}_{ab}, h)}{\partial \mathbf{r}_b}. \tag{3.13}$$

These properties will be utilised throughout this chapter.

3.2 Euler Flow

Using just these interpolation basics, we can now proceed to derive the SPH equations for Euler fluid flow.

3.2.1 Continuity Equation

The continuity equation is a statement of the conservation of mass. While this equation can be transformed into SPH for the purpose of evolving particle densities, this is not required. When particle mass is held fixed, the conservation of mass within the system is inherently preserved. Instead, densities can be calculated anew using

$$\rho_a = \sum_b m_b W_{ab} \tag{3.14}$$

which follows straight from equation 3.5, the definition of SPH interpolation. To further simplify the presentation of the equations, the notation $W_{ab} = W(\mathbf{r}_{ab}, h)$ will be used.

The continuity equation may indeed still be used, instead. Using the definition of the SPH derivative, in this case from equation 3.10, we can transform

$$\frac{d\rho}{dt} = -\rho \nabla \cdot \mathbf{v} \tag{3.15}$$

to become

$$\frac{d\rho_a}{dt} = \sum_b m_b \mathbf{v}_{ab} \cdot \nabla_a W_{ab}, \tag{3.16}$$

where we use the notational shorthand $\mathbf{v}_{ab} = \mathbf{v}_a - \mathbf{v}_b$ similar to that introduced previously.

3.2.2 Momentum Equations

The momentum equations can be derived through a Lagrangian approach using only the density definition from equation 3.14. We begin with the Lagrangian for the SPH system

$$L(\mathbf{r}, \mathbf{v}) = \frac{1}{2} \sum_b m_b \mathbf{v}_b^2 - \sum_b m_b u_b \tag{3.17}$$

which is the sum of kinetic and internal energies from each particle. Here the internal energy is expressed per unit mass.

Next, we use equation 3.17 with the Euler-Lagrange equations,

$$\frac{d}{dt}\frac{\partial L}{\partial \mathbf{v}_a} - \frac{\partial L}{\partial \mathbf{r}_a} = 0. \tag{3.18}$$

The first term simplifies easily to

$$\frac{d}{dt}\frac{\partial L}{\partial \mathbf{v}_a} = m_a \frac{d\mathbf{v}_a}{dt}. \tag{3.19}$$

For the second term, the internal energy is rewritten into a more convienient form. From the first law of thermodynamics,

$$TdS = dU - PdV, \tag{3.20}$$

where T, S, P, and V are as usual the temperature, entropy, pressure and volume. As the system is dissipationless, the change in entropy is zero, and rewriting dV in terms of density, we can achieve

$$du = \frac{P}{\rho^2} d\rho, \tag{3.21}$$

and by associating this equation with the expression for internal enegy in terms of its natural variables

$$du = \left(\frac{\partial u}{\partial \rho}\right)_s d\rho + \left(\frac{\partial u}{\partial s}\right)_\rho ds, \tag{3.22}$$

we can obtain

$$\left(\frac{\partial u}{\partial \rho}\right)_s = \frac{P}{\rho^2}. \tag{3.23}$$

Moving back to the second term in the Lagrangian, we can use this along with the definition of the density to obtain

$$\begin{aligned}
\frac{\partial L}{\partial \mathbf{r}_a} &= -\frac{\partial}{\partial \mathbf{r}_a}\left(\sum_b m_b u_b\right) \\
&= -\sum_b m_b \frac{\partial u_b}{\partial \rho_b}\frac{\partial \rho_b}{\partial \mathbf{r}_a} \\
&= -\sum_b m_b \frac{P_b}{\rho_b^2}\frac{\partial \rho_b}{\partial \mathbf{r}_a} \\
&= -\sum_b m_b \frac{P_b}{\rho_b^2}\sum_c m_c \nabla_a W_{bc}.
\end{aligned} \tag{3.24}$$

Here we have a double summation over the particles. The gradient will pluck out particle a from the outer summation, and seperately a from the inner summation.

This gives

$$\begin{aligned}\frac{\partial L}{\partial \mathbf{r}_a} &= -\sum_b m_b \frac{P_b}{\rho_b^2} \sum_c m_c \nabla_a W_{bc} \delta_{ba} - \sum_b m_b \frac{P_b}{\rho_b^2} \sum_c m_c \nabla_a W_{cb} \delta_{ca} \\ &= -m_a \frac{P_a}{\rho_a^2} \sum_c m_c \nabla_a W_{ac} - \sum_b m_b \frac{P_b}{\rho_b} m_a \nabla_a W_{ab} \\ &= -m_a \sum_b m_b \left(\frac{P_a}{\rho_a^2} + \frac{P_b}{\rho_b^2} \right) \nabla_a W_{ab}. \end{aligned} \tag{3.25}$$

Putting the two halves together, the Euler-Lagrange equations become

$$\frac{d\mathbf{v}_a}{dt} = -\sum_b m_b \left(\frac{P_a}{\rho_a} + \frac{P_b}{\rho_b} \right) \nabla_a W_{ab}, \tag{3.26}$$

giving us our equation for momentum evolution.

This equation will conserve both linear and angular momentum. It satisfies Newton's third law, as the force from particle b on a will be equal and opposite that of a on b. Angular momentum can be seen to be conserved by taking the time derivative of the total angular momentum of the system,

$$\begin{aligned}\frac{d}{dt} \left(\sum_a \mathbf{r}_a \times m\mathbf{v}_a \right) &= \sum_a m_a \left(\mathbf{r}_a \times \frac{d\mathbf{v}_a}{dt} \right) \\ &= \sum_a \sum_b m_a m_b \left(\frac{P_a}{\rho_a^2} + \frac{P_b}{\rho_b^2} \right) \left(\mathbf{r}_a \times (\mathbf{r}_a - \mathbf{r}_b) \right) F_{ab} \\ &= -\sum_a \sum_b m_a m_b \left(\frac{P_a}{\rho_a^2} + \frac{P_b}{\rho_b^2} \right) \mathbf{r}_a \times \mathbf{r}_b F_{ab} \end{aligned} \tag{3.27}$$

This double summation is antisymmetric in a and b. Expanding the list of terms, each particle pair will appear twice - once as an addition and once as a subtraction, cancelling each other out. Thus, the whole expression is zero, and angular momentum

is conserved.

This equation can more simply be derived by applying the interpolation derivative rule to the identity

$$\frac{\nabla P}{\rho} = \nabla\left(\frac{P}{\rho}\right) + \frac{P}{\rho^2}\nabla\rho \tag{3.28}$$

which produces equation 3.26.

3.2.3 Energy Equation

To simulate the thermodynamics of the system properly, the internal energy of each particle needs to be updated in time. This can be done by either calculating the rate of change of the the internal energy, the thermokinetic energy, or the entropy of each particle.

Evolving the internal energy is straightforward. Taking the thermodynamic relation from equation 3.21, we can write

$$\frac{du}{dt} = \frac{P}{\rho^2}\frac{d\rho}{dt}. \tag{3.29}$$

Taking the form of the continuity equation in equation 3.16, we obtain

$$\frac{du_a}{dt} = \frac{P_a}{\rho_a^2}\sum_b m_b \mathbf{v}_{ab}\cdot\nabla_a W_{ab}. \tag{3.30}$$

The thermokinetic energy is the sum of internal and kinetic energies,

$$e = \tfrac{1}{2}\mathbf{v}^2 + u, \tag{3.31}$$

given per unit mass. Taking the time derivative,

$$\frac{dc}{dt} = \mathbf{v} \cdot \frac{dv}{dt} + \frac{du}{dt}, \tag{3.32}$$

we can substitute in equations 3.26 and 3.30 to obtain

$$\frac{dc_a}{dt} = -\sum_b m_b \left(\frac{P_a}{\rho_a^2} \mathbf{v}_b + \frac{P_b}{\rho_b^2} \mathbf{v}_a \right) \cdot \nabla_a W_{ab}. \tag{3.33}$$

As should be expected, the total energy change of the system is zero, since this relation is antisymmetric in a and b. From the thermokinetic energy, the internal energy can be recalculated from equation 3.31 by subtracting out the kinetic energy.

Thirdly, the entropy can be used. For an adiabatic system, we associate the entropy with an entropic function $A(s)$, given from the thermodynamic relation

$$P = A(s)\rho^\gamma \tag{3.34}$$

for an ideal gas. The internal energy can be evaluated from $A(s)$ using

$$u = \frac{A(s)}{\gamma - 1} \rho^{\gamma - 1}. \tag{3.35}$$

We can see that the time evolution of the entropic function will be zero by

$$\begin{aligned} \frac{dA(s)_a}{dt} &= \frac{d(\gamma - 1)u_a}{d\rho_a^{\gamma - 1}} \\ &= \frac{(\gamma - 1)}{\rho_a^{\gamma - 1}} \left(\frac{du_a}{dt} - \frac{P_a}{\rho_a^2} \frac{d\rho_a}{dt} \right) \\ &= 0. \end{aligned} \tag{3.36}$$

By monitoring the entropy of the system instead of the energy, it becomes very easy

to control any sources of entropy, which will only arise from dissipative effects or sources outside the system.

3.3 Variable Resolution Smoothing Lengths

So far the SPH equations have been presented with the smoothing length being constant over space and time. This can lead to problems when the length scale of a system changes substantially. In the extreme case, particles in regions of sufficiently low density can become isolated, which breaks the method. Conversely, when densities become extreme, such regions become increasingly oversmoothed.

To alleviate the resolution concerns outlined, smoothing lengths may be set individually per particle. The smoothing length of each particle is allowed to vary with time so that it encloses a roughly constant number of neighbouring particles.

The methodologies which have emerged to adapt smoothing lengths can be considered as divided into two seperate styles. The first uses criteria based on the density to adapt the smoothing length. For example, the method of Benz [6] takes the derivative of the scaling law

$$h_a = \eta \left(\frac{m_a}{\rho_a} \right)^{1/\upsilon} \tag{3.37}$$

to obtain

$$\frac{dh_a}{dt} = -\frac{1}{\upsilon} \frac{h_a}{\rho_a} \frac{d\rho_a}{dt} \tag{3.38}$$

where υ is the dimension of the system and η is a constant ~ 1.5. The time derivative on ρ can be replaced by the continuity equation 3.16. In this way, h is evolved as a system variable.

The second methodology uses the number of neighbours to adjust the smoothing length. When the neighbour count is below the ideal, h is increased, and conversely so when the neighbour count is too high. The method of Hernquist and Katz [22] defines the new smoothing length as

$$h_a^{\text{new}} = \frac{h_a^{\text{old}}}{2}\left[1 + \left(\frac{N_{\text{ideal}}}{N_{\text{neigh}}}\right)^{1/3}\right] \tag{3.39}$$

which predicts the next smoothing length as the average of the current smoothing length and the smoothing length implied by the current number of neighbours.

Both of these methods aim to keep the number of neighbours roughly constant over time. As they are only estimation methods, strict control over the neighbour count is not guaranteed, but their efficiency is appealing. In fact, a third class of methods exist, those which are not estimation methods, but which provide smoothing lengths to keep the neighbour count precisely constant. Such a method was developed during the course of this research, and is described further in section 5.2.2.

3.3.1 ∇h Correction Terms

Allowing the smoothing lengths to vary in space complicates the SPH fluid equations. Gradients of the smoothing kernel now produce additional $\partial W/\partial h$ terms, called the ∇h terms.

Nelson and Papaloizou [34] derived a form of these terms using a symmetrized kernel

$$W'_{ab} = \tfrac{1}{2}[W'_{ab}(h_a) + W'_{ab}(h_b)] \tag{3.40}$$

in the definition of interpolation, and obtained the additional terms,

$$
\begin{aligned}
&- \frac{1}{4}\hat{\mathbf{r}}_a \sum_b m_b \left(\frac{P_a}{\rho_a^2} + \frac{P_b}{\rho_b^2} \right) \frac{\partial W_{ab}(h_a)}{\partial h_a} \\
&+ \frac{1}{4} \sum_b \hat{\mathbf{r}}_b \delta_{ab_m} \sum_c \frac{m_b m_c}{m_a} \left(\frac{P_b}{\rho_b^2} + \frac{P_c}{\rho_c^2} \right) \frac{\partial W_{bc}(h_b)}{\partial h_b},
\end{aligned}
\tag{3.41}
$$

in the momentum equation. Here $\hat{\mathbf{r}}_k = \mathbf{r}_{kk_m} / |r_{kk_m}|$ where the subscript k_m denotes the most distant neighbour of k. Neglecting these terms leads to an unphysical entropy generation [21, 35, 48, 1], but even so, their inclusion has not found widespread use due to their complicated form.

3.3.2 A Fully Conservative Formulation

Springel and Hernquist [50] developed a new formulation which accounts for the ∇h terms in a simple manner. The canonical variables of the Lagrangian system are extended to include the smoothing lengths, $\mathbf{q} = (\mathbf{r}_1, \ldots, \mathbf{r}_N, h_1, \ldots, h_N)$. Smoothing lengths are selected to encompass a fixed amount of mass (which translates to a constant neighbour count for particles of uniform and fixed mass), by the relation

$$
\frac{4\pi}{3} h_a^3 \rho_a = M \tag{3.42}
$$

where M is the desired enclosed mass. This places a constraint on the Euler-Lagrange equations of the form

$$
\phi_a(\mathbf{q}) \equiv \frac{4\pi}{3} h_a^3 \rho_a - M = 0. \tag{3.43}
$$

Then from the Euler-Lagrange equations,

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\mathbf{q}}_a} - \frac{\partial L}{\partial q_a} = \sum_b^{2N} \lambda_b \frac{\partial \phi_b}{\partial \mathbf{q}_a}, \tag{3.44}$$

we can find the Lagrange multipliers, λ_a, to be

$$\lambda_b = \frac{3}{4\pi}\frac{m_a}{h_a^3}\frac{P_a}{\rho_a^2}\left[1 + \frac{3\rho_a}{h_a}\left(\frac{\partial \rho_a}{\partial h_a}\right)^{-1}\right]^{-1}. \tag{3.45}$$

This yields equations of motion

$$\frac{d\mathbf{v}_a}{dt} = -\sum_b m_b \left[f_a \frac{P_a}{\rho_a^2}\boldsymbol{\nabla}_a W_{ab}(h_a) + f_b \frac{P_b}{\rho_b^2}\boldsymbol{\nabla}_b W_{ab}(h_b)\right] \tag{3.46}$$

where

$$f_a = \left(1 + \frac{h_a}{3\rho_a}\frac{\partial \rho_a}{\partial h_a}\right)^{-1} \tag{3.47}$$

$$= \left(1 + \frac{h_a}{3\rho_a}\sum_b m_b \frac{\partial W_{ab}(h_a)}{\partial h_a}\right)^{-1} \tag{3.48}$$

Monaghan [29] also derived this form of equation by realising that

$$\frac{\partial \rho_b}{\partial \mathbf{r}_a} = \sum_c m_c \left(\boldsymbol{\nabla}_a W_{bc}(h_b) + \frac{\partial W_{bc}(h_b)}{\partial h_b}\frac{\partial h_b}{\partial \rho_b}\frac{\partial \rho_b}{\partial \mathbf{r}_a}\right). \tag{3.49}$$

By collecting the $\partial\rho_b/\partial\mathbf{r}_a$ terms, we obtain

$$\frac{\partial \rho_b}{\partial \mathbf{r}_a} = \frac{1}{\Omega_b}\sum_c m_c \boldsymbol{\nabla}_a W_{bc}(h_b) \tag{3.50}$$

where

$$\Omega_a = \left(1 - \frac{\partial h_a}{\partial \rho_a}\sum_b m_b \frac{\partial W_{ab}(h_a)}{\partial h_a}\right). \tag{3.51}$$

Placing equation 3.50 into 3.24, the equations of motion of the form

$$\frac{d\mathbf{v}_a}{dt} = -\sum_b m_b \left(\frac{1}{\Omega_a} \frac{P_a}{\rho_a^2} \nabla_a W_{ab}(h_a) + \frac{1}{\Omega_b} \frac{P_b}{\rho_b^2} \nabla_a W_{ab}(h_b) \right) \tag{3.52}$$

emerge. We can see that if the smoothing length is adapted as in equation 3.37, then

$$\frac{\partial h_a}{\partial \rho_a} = -\frac{1}{v} \frac{h_a}{\rho_a} \tag{3.53}$$

which shows that $\Omega_a^{-1} \equiv f_a$. This can be understood more clearly by recognizing that this smoothing length relation and Springel and Hernquist's constraint have the same meaning.

These factors appear in the energy equation as

$$\frac{du_a}{dt} = \frac{1}{\Omega_a} \frac{P_a}{\rho_a^2} \sum_b m_b \mathbf{v}_{ab} \cdot \nabla_a W_{ab}(h_a). \tag{3.54}$$

3.4 Artificial Viscosity

It is necessary to include an artificial viscosity in order to model shocks correctly. The smallest structures that can be modeled with SPH are on the order of the smoothing length, which causes the width of shocks to broaden out to this length. Post shock oscillations are similar in width to the shock, but due to broadening of the shock they become unphysically macroscopic. To remove these post shock oscillations, an artificial viscosity is introduced into the fluid equations, appearing in the form of a pressure like term, which dissipates kinetic energy into internal energy.

3.4.1 von Neumann-Richtymer Analogy; the "Standard" Form

Monaghan and Gingold [30] introduced an artifical viscosity which commonly became referred to as the "standard" viscosity due to its popularity. Its form is analagous to the von Neumann-Richtmyer viscosity used in finite-difference methods. The artificial viscosity, Π_{ab}, is given by

$$\Pi_{ab} = \begin{cases} -\dfrac{\alpha \bar{c}_{ab} \mu_{ab} + \beta \mu_{ab}^2}{\bar{\rho}_{ab}} & \mathbf{v}_{ab} \cdot \mathbf{r}_{ab} < 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.55}$$

with

$$\mu_{ab} = \frac{h \mathbf{v}_{ab} \cdot \mathbf{r}_{ab}}{\mathbf{r}_{ab}^2 + \epsilon h}. \tag{3.56}$$

The parameter $\epsilon \sim 0.01$ is used to guard against numerical divergences, and α and β are constants typically with values $\alpha = 1$ and $\beta = 2\alpha$. The speed of sound, c, is given by $c_a = \sqrt{\gamma P_a / \rho_a}$ for an ideal gas, with $\bar{c}_{ab} = \frac{1}{2}(c_a + c_b)$ being the average speed of sound between the two particles. When using individual smoothing lengths, h is similarly averaged to $\bar{h}_{ab} = \frac{1}{2}(h_a + h_b)$. As the artificial viscosity is intended only to provide the dissipation required at a shock, it is only applied when the condition $\mathbf{v}_{ab} \cdot \mathbf{r}_{ab} < 0$ is fulfilled, which occurs for approaching particles.

The first term involving the speed of sound in the artificial viscosity represents a bulk viscosity. The second term is present to simulate a shear viscosity, and its inclusion is necessary to prevent inter-particle penetration during high Mach number collisions and shocks.

The artificial viscosity is added to the momentum equation as

$$\left(\frac{d\mathbf{v}_a}{dt}\right)_{\text{visc}} = -\sum_b m_b \Pi_{ab} \nabla_a W_{ab}. \tag{3.57}$$

To find the corresponding increase in internal energy from the viscous dissipation, we examine the equation for change in total thermokinetic energy of the system,

$$\frac{dE}{dt} = \sum_a m_a \frac{de_a}{dt}. \tag{3.58}$$

Naturally, this will equal zero, but from it we can isolate the change in kinetic energy from the artificial viscosity to find the form for $(du_a/dt)_{\text{visc}}$. By recognising that the non-dissipative changes already balance, this yields

$$\begin{aligned}
\sum_a m_a \left(\frac{du_a}{dt}\right)_{\text{visc}} &= \sum_a m_a \mathbf{v}_a \left(\frac{d\mathbf{v}_a}{dt}\right)_{\text{visc}} \\
&= \sum_a m_a \sum_b m_b \Pi_{ab} \mathbf{v}_a \cdot \mathbf{r}_{ab} F_{ab} \\
&= \frac{1}{2} \sum_a \sum_b m_a m_b \Pi_{ab} \mathbf{v}_a \cdot \mathbf{r}_{ab} F_{ab} + \frac{1}{2} \sum_b \sum_a m_b m_a \Pi_{ba} \mathbf{v}_b \cdot \mathbf{r}_{ba} F_{ba} \\
&= \frac{1}{2} \sum_a \sum_b m_a m_b \Pi_{ab} \mathbf{v}_a \cdot \mathbf{r}_{ab} F_{ab} - \frac{1}{2} \sum_b \sum_a m_b m_a \Pi_{ba} \mathbf{v}_b \cdot \mathbf{r}_{ab} F_{ab} \\
&= \frac{1}{2} \sum_a m_a \sum_b m_b \Pi_{ab} \mathbf{v}_{ab} \cdot \nabla_a W_{ab}
\end{aligned} \tag{3.59}$$

From this, we see that

$$\left(\frac{du_a}{dt}\right)_{\text{visc}} = \frac{1}{2} \sum_b m_b \Pi_{ab} \mathbf{v}_{ab} \cdot \nabla_a W_{ab} \tag{3.60}$$

3.4.2 Riemann Solver Analagy; the "New" Standard

A new form for artificial viscosity was introduced by Monaghan in 1997 [28] based on analogy from Riemann solvers. It uses the signal velocity between two particles a and b which are treated as left and right Riemann states. This artificial viscosity has the form

$$\Pi_{ab} = \frac{-\alpha v_{ab}^{\text{sig}} \mathbf{v}_{ab} \cdot \hat{\mathbf{r}}_{ab}}{\overline{\rho}_{ab}} \tag{3.61}$$

where $\hat{\mathbf{r}}_{ab} = \mathbf{r}_{ab} / |\mathbf{r}_{ab}|$. This is similarly only applied during compression, when $\mathbf{v}_{ab} \cdot \mathbf{r}_{ab} < 0$ is satisfied. The term $\mathbf{v}_{ab} \cdot \hat{\mathbf{r}}_{ab}$ represents the velocity difference taken along the line joining the particles, since during shocks only the component perpindicular to the shock is expected to change. The signal velocity, v^{sig}, is given by

$$v_{ab}^{\text{sig}} = c_a + c_b - \beta \mathbf{v}_{ab} \cdot \hat{\mathbf{r}}_{ab} \tag{3.62}$$

which is the speed of two approaching sound waves adjusted for the relative motion of the two particles.

This will increase thermokinetic energy at a rate of

$$\left(\frac{de_a}{dt}\right)_{\text{visc}} = \sum_b m_b v_{ab}^{\text{sig}} \frac{e_a^* - e_b^*}{\overline{\rho}_{ab}} \hat{\mathbf{r}}_{ab} \cdot \boldsymbol{\nabla}_a W_{ab} \tag{3.63}$$

where $c_a^* = \frac{1}{2}\alpha(\mathbf{v}_a \cdot \hat{\mathbf{r}}_{ab})^2 + \alpha_u u_a$. The kinetic energy term in this definition is projected along the line joining the particles. Seperate viscosity parameters have been used for the kinetic and internal energy terms.

From the rate of change of the thermokinetic energy (equation 3.32), we can substitute equations 3.63 and 3.57 to obtain a form for the rate of change of internal energy given as

$$\left(\frac{du_a}{dt}\right)_{\text{visc}} = \sum_b m_b \frac{v_{ab}^{\text{sig}}}{\bar{\rho}_{ab}} \left(\alpha_u(u_a - u_b) - \frac{\alpha}{2}(\mathbf{v}_{ab}\cdot\hat{\mathbf{r}}_{ab})^2\right)\hat{\mathbf{r}}_{ab}\cdot\boldsymbol{\nabla}_a W_{ab}. \tag{3.64}$$

Of the two terms in this equation, the α term provides the increase in internal energy from the viscous dissipation, while the α_u acts as an artificial thermal conductivity, transferring internal energy between particles.

3.4.3 Viscosity Limiters

The inclusion of artificial viscosity is necessary to properly model shock phenomena, however its dissipative effects can be too great when applied universally to the system. Viscosity limiting strategies have been developed to address this issue.

Balsara [2] multiplied the viscosity tensor, Π_{ab}, by a factor, f_{ab}, to suppress the viscosity in pure shear flows. This factor is given by $f_{ab} = \frac{1}{2}(f_a + f_b)$ with

$$f_a = \frac{|\boldsymbol{\nabla}\cdot\mathbf{v}_a|}{|\boldsymbol{\nabla}\cdot\mathbf{v}_a| + |\boldsymbol{\nabla}\times\mathbf{v}_a| + \eta\bar{c}_{ab}/h}, \tag{3.65}$$

where a safety term is added to the denominator to avoid numerical divergences. In pure compression regions such as shocks, $|\boldsymbol{\nabla}\times\mathbf{v}_a| = 0$ and the full force of the artificial viscosity is applied. During pure shear flows however, $|\boldsymbol{\nabla}\cdot\mathbf{v}_a| = 0$ and the viscosity is restrained.

A different approach by Morris and Monaghan [33] is to use time dependent artificial parameters set individually per particle, replacing α in the artificial viscosity equations with $\frac{1}{2}(\alpha_a + \alpha_b)$. They are evolved as

$$\frac{d\alpha_a}{dt} = -\frac{(\alpha_a - \alpha_{\min})}{\tau} + S_a, \tag{3.66}$$

where the first term decays the strength of the artificial viscosity, and S_a is a source term for increasing the strength. The timescale for decay, τ, has been calculated as

$$\tau = \frac{h_a}{C c_a}. \tag{3.67}$$

As the post shock Mach number for strong shocks is $\sqrt{(\gamma-1)/2\gamma} \approx 0.447$ for $\gamma = 5/3$, values for the constant C are typically chosen to be $\sim 0.1-0.2$, which corresponds to the decay being spread over 5 to 2 smoothing length. The decay is limited to a minimum value of $\alpha_{\min}$.

Rosswog *et al.* [44] suggested a source term of

$$S_a = \max\left(-\boldsymbol{\nabla}\cdot\mathbf{v}_a, 0\right)\left(\alpha_{\max} - \alpha_a\right). \tag{3.68}$$

This increases the strength of the viscosity in regions of compression, which makes it function similar to Balsara's limiter. Analagously to the decay, the source contribution is limited to a maximum value of $\alpha_{\max}$.

This strategy may also be employed for the artificial thermal conductivity parameter. For this case, the source term can be calculated as

$$S_a^u = 0.1h\left|\boldsymbol{\nabla}^2 u_a\right| \tag{3.69}$$

as suggested by Price and Monaghan [43]. Using the second derivative is found to be preferable since it responds only to sharp discontinuities in the internal energy.

3.5 Magnetohydrodynamics

While SPH has matured since its inception, its implementation of the equations of magnetohydrodynamics remains wanting. A conservative formulation can easily be

derived for the equations of motion, but it was found that straightforward application of the SPH method to the MHD equations can lead to instabilities related to the non-zero divergence of the magnetic field [37]. When the magnetic pressure is greater than the thermal pressure, negative stress produces a force parallel to the magnetic field and porportional to $\nabla \cdot \mathbf{B}$ which causes particles to clump together.

Simply enforcing the div $\boldsymbol{B}$ constraint does not resolve the problem. The conservative formulation calculates gradients which do not equal zero even for constant functions. Several strategies have been implemented to combat this instability, some with more success than others. Two approaches have emerged which have proven the most effective and popular, and will be discussed.

Of course, even though the tensile instability may exist for zero $\nabla \cdot \mathbf{B}$, it still remains important to fulfill this constraint. Price and Monaghan examined parabolic and hyperbolic divergence cleaning techniques [43], which were found to yield good results for two dimensional test problems, but which performed poorly for real three dimensional applications such as star formation [41]. Present approaches formulate the magnetic field in terms of Euler potentials, and will be discussed.

To round out things out, forms for the induction equation will be discussed, along with equations for implementing an artificial magnetic resistivity.

3.5.1 Momentum Equation

The first form of the SPMHD momentum equations were derived by Phillips and Monaghan [37] using the Maxwell stress tensor, $\mathbf{M}$. This straightforwardly gives

$$\left(\frac{d\mathbf{v}_a}{dt}\right)_{\text{mag}} = \sum_b m_b \left[\frac{\mathbf{M}_a}{\rho_a^2} + \frac{\mathbf{M}_b}{\rho_b^2}\right] \cdot \nabla_a W_{ab}, \tag{3.70}$$

or in component form

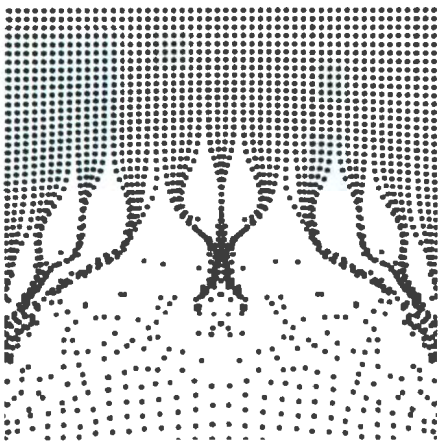

Figure 3.1: Example of particles clumping together under negative stress from the instability in the conservative SPMHD momentum equation.

$$\left(\frac{d\mathbf{v}_a^i}{dt}\right)_{\text{mag}} - -\frac{1}{2\mu_0}\sum_b m_b\left[\frac{B_a^2}{\rho_a^2}+\frac{B_b^2}{\rho_b^2}\right]\frac{\partial W_{ab}}{\partial x^i} + \frac{1}{\mu_0}\sum_b m_b\left[\frac{B_a^i B_a^j}{\rho_a^2}+\frac{B_b^i B_b^j}{\rho_b^2}\right]\frac{\partial W_{ab}}{\partial x^j}. \tag{3.71}$$

However, as discussed previously, this equation produces an instability when the magnetic pressure is greater than the thermal pressure. This can be seen more clearly by examining the two summation terms in 3.71. While the isotropic terms in the first summation are guaranteed to yield positive stress, no such guarantee can be made for the anisotropic terms. This is the source of the clumping instability, for which an example is presented in figure 3.1.

3.5.1.1 Børve $\nabla \cdot \mathbf{B}$ Subtraction

The approach by Børve *et al.* [8] has proved simple yet effective in preventing the instability. Their idea is straightforward: calculate the force that arises from non-zero

$\nabla \cdot \mathbf{B}$ and subtract it from the net force. As previously mentioned, the force in the instability is aligned along the magnetic field and is proportional to $\nabla \cdot \mathbf{B}$, so they convert the expression $\mathbf{B}\nabla \cdot \mathbf{B}$ into SPH format yielding

$$\left(\frac{d\mathbf{v}_a}{dt}\right)_{\text{div B}} = -\mathbf{B}_a \sum_b m_b \left(\frac{\mathbf{B}_a}{\Omega_a \rho_a^2} + \frac{\mathbf{B}_b}{\Omega_b \rho_b^2}\right) \cdot \nabla_a W_{ab}. \tag{3.72}$$

The Ω terms have been included to account for spatially varying smoothing lengths, which were not included in the original formulation. Note that including this term breaks the momentum conservation of the system, but this sacrifice is considered preferable to the clumping instability.

3.5.1.2 Morris Approach

Morris modified the anisotropic part of the momentum equation to use a differencing formula, which vanishes for constant stress [32]. The isotropic and thermal pressure terms are computed as previous. This leads to a form

$$\begin{aligned}\left(\frac{d\mathbf{v}_a}{dt}\right)_{\text{mag}} = &-\frac{1}{2\mu_0}\sum_b m_b \left[\frac{B_a^2}{\Omega_a \rho_a^2}\nabla_a W_{ab}(h_a) + \frac{B_b^2}{\Omega_b \rho_b^2}\nabla_a W_{ab}(h_b)\right] \\ &+ \frac{1}{\mu_0}\sum_b m_b \left[\frac{B_a^i B_a^j - B_b^i B_b^j}{\rho_a \rho_b}\right] \cdot \overline{\nabla_a W_{ab}},\end{aligned} \tag{3.73}$$

which has been modified from his initial description so that the isotropic terms account for variable smoothing lengths. The Ω factors are not applied to the anisotropic terms, since they use a different definition for the SPH derivative, and instead the averaged kernel is used.

This form for the momentum equation breaks the momentum conservation of the system, but as with the Borve approach, removal of the instability has more

importance. The Morris formulation is found to effectively remove the clumping instability.

3.5.2 Induction Equation

As with the momentum equation, the SPH form of the induction equation can easily be derived, yielding

$$\frac{d\mathbf{B}_a}{dt} = -\frac{1}{\rho_a}\sum_b m_b \left[\mathbf{v}_{ab}(\mathbf{B}_a \cdot \boldsymbol{\nabla}_a W_{ab}) - \mathbf{B}_a(\mathbf{v}_{ab} \cdot \boldsymbol{\nabla}_a W_{ab})\right] \tag{3.74}$$

or alternatively,

$$\frac{d}{dt}\left(\frac{\mathbf{B}_a}{\rho_a}\right) = -\frac{1}{\rho_a^2}\sum_b m_b \mathbf{v}_{ab}(\mathbf{B}_a \cdot \boldsymbol{\nabla}_a W_{ab}). \tag{3.75}$$

These forms, however, do not account for variable smoothing lengths. To incorporate their effects into the induction equation, we first begin with the continuity equation accounting for the Ω terms, given as

$$\frac{d\rho_a}{dt} = \frac{1}{\Omega_a}\sum_b \mathbf{v}_{ab} \cdot \boldsymbol{\nabla}_a W_{ab}. \tag{3.76}$$

By expanding the left hand side of equation 3.75 using the identity $\boldsymbol{\nabla}(A/\rho) = \boldsymbol{\nabla}A/\rho - A/\rho^2\boldsymbol{\nabla}\rho$, and using 3.76, we can see that

$$\begin{aligned}\frac{d\mathbf{B}_a}{dt} &= -\frac{1}{\rho_a}\sum_b m_b \mathbf{v}_{ab}(\mathbf{B}_a \cdot \boldsymbol{\nabla}_a W_{ab}) + \frac{\mathbf{B}_a}{\rho_a}\frac{d\rho}{dt} \\ &= -\frac{1}{\rho_a}\sum_b m_b \left[\mathbf{v}_{ab}(\mathbf{B}_a \cdot \boldsymbol{\nabla}_a W_{ab}) - \frac{1}{\Omega_a}\mathbf{B}_a(\mathbf{v}_{ab} \cdot \boldsymbol{\nabla}_a W_{ab})\right].\end{aligned} \tag{3.77}$$

Since the induction equation should equal zero for one dimensional systems, this

imposes the requirement that the two terms cancel. Thus, the correct form should contain the Ω factors on both terms, producing

$$\frac{d\mathbf{B}_a}{dt} = -\frac{1}{\Omega_a \rho_a} \sum_b m_b \left[\mathbf{v}_{ab} (\mathbf{B}_a \cdot \boldsymbol{\nabla}_a W_{ab}) - \mathbf{B}_a (\mathbf{v}_{ab} \cdot \boldsymbol{\nabla}_a W_{ab}) \right], \tag{3.78}$$

and

$$\frac{d}{dt}\left(\frac{\mathbf{B}_a}{\rho_a}\right) = -\frac{1}{\Omega_a \rho_a^2} \sum_b m_b \mathbf{v}_{ab} (\mathbf{B}_a \cdot \boldsymbol{\nabla}_a W_{ab}). \tag{3.79}$$

3.5.3 Ohmic Dissipation

Price and Monaghan investigated incorporating an artificial resitivity into the induction equation to simulate Ohmic dissipation [42, 43]. A process similar to that of Monaghan's artificial viscosity based on analogy to Riemann solvers [28] was used to derive their form.

As before, the rate of change of energy per particle from dissipative sources will be given as

$$\left(\frac{de_a}{dt}\right)_{\text{diss}} = \sum_b m_b \frac{v_{ab}^{\text{sig}}}{\overline{\rho}_{ab}} (e_a^* - e_b^*) \hat{\mathbf{r}}_{ab} \cdot \boldsymbol{\nabla}_a W_{ab}, \tag{3.80}$$

but with the magnetic component of the total energy included in the e^* terms as

$$e_a^* = \frac{\alpha}{2} \mathbf{v}_a^2 + \alpha_u u_a + \alpha_B \frac{\mathbf{B}_a^2}{2\mu_0 \overline{\rho}_{ab}}. \tag{3.81}$$

This differs from the Monaghan's artificial viscosity in that the velocity is not taken along the line between the two particles, and neither is the magnetic field. This deviation was motivated by the assumption that the div $\boldsymbol{B}$ constraint may not be upheld exactly, and components of the velocity and magnetic field not perpendicular

to a shock may be involved.

The rate of change of internal energy due to viscous and resistive heating can be found then to be

$$\left(\frac{du_a}{dt}\right)_{\text{diss}} = -\sum_b m_b \frac{v_{ab}^{\text{sig}}}{\overline{\rho}_{ab}} \left[\frac{\alpha}{2}(\mathbf{v}_a - \mathbf{v}_b)^2 + \alpha_u(u_a - u_b) \quad + \frac{\alpha_B}{2\mu_0 \overline{\rho}_{ab}}(\mathbf{B}_a - \mathbf{B}_b)^2\right] \hat{\mathbf{r}}_{ab} \cdot \boldsymbol{\nabla}_a W_{ab}, \tag{3.82}$$

which has been formulated so that the viscous and resistive components will be a positive contribution to the internal energy.[1]

This requires the dissipation terms to have form

$$\left(\frac{d\mathbf{v}_a}{dt}\right)_{\text{diss}} = \sum_b m_b \frac{\alpha v_{ab}^{\text{sig}}}{\overline{\rho}_{ab}}(\mathbf{v}_a - \mathbf{v}_b)\hat{\mathbf{r}}_{ab} \cdot \boldsymbol{\nabla}_a W_{ab}, \tag{3.83}$$

$$\left(\frac{d\mathbf{B}_a}{dt}\right)_{\text{diss}} = \rho_a \sum_b m_b \frac{\alpha_B v_{ab}^{\text{sig}}}{\overline{\rho}_{ab}}(\mathbf{B}_a - \mathbf{B}_b)\hat{\mathbf{r}}_{ab} \cdot \boldsymbol{\nabla}_a W_{ab}. \tag{3.84}$$

As before, the artificial viscosity is only applied during compression, when $\mathbf{v}_{ab} \cdot \hat{\mathbf{r}}_{ab} < 0$, but the artificial resistivity is applied universally across the system. It is important to mention that this equation for artificial viscosity no longer conserves angular momentum (though linear momentum remains conserved), which occurs since the viscosity is no longer directed along the line between the two particles.

When applying the artificial viscosity and resistivity for magnetohydrodynamic systems, the signal velocity should be changed to use the speed of the fast MHD wave instead of the speed of sound. This is given by

[1] This can be seen more clearly by recognising that $\boldsymbol{\nabla}_a W_{ab} = \mathbf{r}_{ab} F_{ab}$ where $F_{ab} \leq 0$, and $\hat{\mathbf{r}}_{ab} \cdot \mathbf{r}_{ab}\, F_{ab} = |r_{ab}|\, F_{ab}$.

$$v_a^{\rm sig} = \frac{1}{\sqrt{2}}\left[\left(c_a^2 + \frac{B_a^2}{\mu_0\rho_a}\right) + \sqrt{\left(c_a^2 + \frac{B_a^2}{\mu_0\rho_a}\right)^2 - 4\frac{c_a^2\left(\mathbf{B}_a\cdot\hat{\mathbf{r}}_{ab}\right)^2}{\mu_0\rho_a}}\right]^{1/2}, \tag{3.85}$$

with the signal velocity becoming

$$v_{ab}^{\rm sig} = v_a + v_b - \beta\mathbf{v}_{ab}\cdot\hat{\mathbf{r}}_{ab}. \tag{3.86}$$

The strength of the artificial resistivity can be regulated using individual time varying parameters in the same manner as the artificial viscosity and thermal conduction parameters (see section 3.4.3.) The source term is chosen to be

$$S_B = \max\left(\frac{|\boldsymbol{\nabla}\times\mathbf{B}|}{\sqrt{\mu_0\rho}}, \frac{|\boldsymbol{\nabla}\cdot\mathbf{B}|}{\sqrt{\mu_0\rho}}\right). \tag{3.87}$$

The decay timescale is similarly adjusted to use the magnetic wave speed described above instead of the speed of sound. Since the definition in equation 3.85 depends upon particle b, the largest value of the computed signal velocities is used, to give

$$\tau = \frac{h_a}{C\max(v_a)}. \tag{3.88}$$

3.5.4 $\boldsymbol{\nabla}\cdot\mathbf{B} = 0$ Constraint : Euler Potentials

Rosswog and Price used Euler potentials to represent the magnetic field in their SPMHD code [45], which satisfies the div $\boldsymbol{B}$ constraint exactly. The magnetic field in terms of the Euler potentials is given by

$$\mathbf{B} = \boldsymbol{\nabla}\alpha\times\boldsymbol{\nabla}\beta. \tag{3.89}$$

Taking the divergence of the above definition shows that the div $\boldsymbol{B}$ constraint is upheld by construction. Geometrically, the magnetic field as defined by the Euler potentials can be considered as the line tangent to surfaces of constant α and constant β [51].

For ideal MHD, the potentials will stay constant in time

$$\frac{d\alpha}{dt} = 0, \quad \frac{d\beta}{dt} = 0. \tag{3.90}$$

Due to this, the variation of the magnetic field only occurs due to the motion of the particle, which corresponds to the frozen field theorem. By using the Euler potential formulism, the induction equations described in section 3.5.2 do not need to be followed.

While the momentum equation can be written to use the Euler potentials, this introduces second derivatives and it is no longer possible to maintain momentum conservation. Thus, the procedure is to use equation 3.89 to obtain the magnetic field, and use it in the standard forms for the momentum equation.

To calculate the gradients of the Euler potentials, the equations

$$\boldsymbol{\nabla}_a^i \alpha_a = \left(\chi^{ij}\right)^{-1} \sum_b m_b(\alpha_b - \alpha_a)\boldsymbol{\nabla}_a^j W_{ab}(h_a), \tag{3.91}$$

$$\boldsymbol{\nabla}_a^i \beta_a = \left(\chi^{ij}\right)^{-1} \sum_b m_b(\beta_b - \beta_a)\boldsymbol{\nabla}_a^j W_{ab}(h_a) \tag{3.92}$$

are used, where

$$\chi^{ij} = \sum_b m_b(\mathbf{r}_b - \mathbf{r}_a)^i \boldsymbol{\nabla}_a^j W_{ab}(h_a). \tag{3.93}$$

The use of Euler potentials does contain some disadvantages. First, given the state of the initial magnetic field, it can be challenging to find Euler potentials which create

that configuration. In addition, certain geometries simply cannot be represented, such as a linked poloidal and toroidal field [52], and it leads to constraints on magnetic helicity [24].

Rosswog and Price incorporated dissipative effects into the Euler potential formulism by evolving the potentials as

$$\left(\frac{d\alpha_a}{dt}\right)_{\rm diss} = \sum_b m_b \frac{\alpha_B v_{ab}^{\rm sig}}{\overline{\rho_{ab}}}(\alpha_a - \alpha_b)\overline{\nabla_a W_{ab}}, \tag{3.94}$$

$$\left(\frac{d\beta_a}{dt}\right)_{\rm diss} = \sum_b m_b \frac{\alpha_B v_{ab}^{\rm sig}}{\overline{\rho_{ab}}}(\beta_a - \beta_b)\overline{\nabla_a W_{ab}}. \tag{3.95}$$

They recognise that this does not rigourously represent Ohmic dissipation, and energy conservation is not strictly observed, but find it satisfactory for one dimensional shock tube tests.

"Sure, slingshot around the sun. If you pick up enough speed you're in time warp."

Leonard "Bones" McCoy

4

Gravity

With the hydrodynamics being modeled using particles, incorporating gravitational forces into the system dynamics is an easy task. This reproduces the well studied gravitational N-body problem, which is to find the gravitational attraction between N bodies. This is labelled a problem because the gravitational force is long range, and each particle needs to interact with the entire system. Two approaches are implemented to compute the gravitational force in this research. One is the straightforward brute force approach, which scales as $O(N^2)$. The other is through the Barnes Hut octree, which scales as $O(N \lg N)$.

4.1 Gravitational Potential

The Newtonian gravitational potential is used, modified with Plummer softening. Between a pair of masses m_a and m_b, this is given by

$$\Phi = -G\frac{m_a m_b}{(r_{ab}^2 + \epsilon^2)^{1/2}}, \tag{4.1}$$

where G is the gravitational constant and r_{ab} is the separation distance between the two masses. Here, ϵ is the softening parameter, a small value added to prevent numerical divergences for close encounters. Its value should be small enough so as to not greatly affect the force outside of those situations. Physically, it amounts to computing the potential as if a small distance was added to the seperation. This potential results in a gravitational force of

$$F_a^G = -G\frac{m_a m_b}{r_{ab}^2 + \epsilon^2}\hat{\mathbf{r}}_{ab} \tag{4.2}$$

directed along the line between the two masses.

4.2 Brute Force Algorithm

The straightforward approach to calculating the gravitational forces is done by directly implementing equation 4.2. Each particle pair in the system is iterated over, with the total force on any one particle being

$$F_a^G = -G\sum_b^N \frac{m_a m_b}{(r_{ab}^2 + \epsilon^2)}\hat{\mathbf{r}}_{ab}. \tag{4.3}$$

Computing this sum for all particles leads this scheme to scale as $O(N^2)$, and is thus rarely used. One benefit to this approach, however, is that momentum will be exactly conserved. The force between each pair of particles is applied equally, thus obeying Newton's third law.

4.3 Barnes–Hut Octree Algorithm

The Barnes Hut (BH) octree algorithm [4] is a method for grouping together the masses of particles so they act as a single entity for the calculation of their gravitational force upon a target particle. The scale of this grouping is based upon their proximity to each other and the distance from the target particle.

A hierarchical grouping scheme is used called an octree (or quadtree for two dimensions). This recursively subdivides a cubic volume of space into eight children cubes, halting once a cube contains one or fewer particles. This forms a tree data structure, with the root node containing the full volume of space and all particles, and leaf nodes containing one particle. Empty nodes need not be stored. The interior nodes of the tree are thus of varying spatial resolution and population count.

To calculate the gravitational force on a particle, we walk down the tree, using an opening criterion to determine how far down a branch of the tree to traverse. Each node in the tree will contain the total mass and center of mass for the particles it contains. Once a node satisfies the opening criterion, the force on the target is calculated using the value stored in that node, and further walking down that branch is discontinued. This saves the computational effort of individually interacting with each member particle of that node.

4.3.1 Opening Critera

The simplest opening criterion as introduced by Barnes and Hut is

$$\theta < \frac{l}{D}. \tag{4.4}$$

Here l represents the width of the cell, and D is the distance from the target to the

center of mass of the cell. A tunable parameter θ is used to control how deep down the tree to traverse. It can be viewed as the tangent of the angle subtended by the width of the node from the perspective of the target particle. This compactly combines node dimension and distance into one parameter, letting nodes grow in size relative to distance. Typically, values chosen are $\theta \sim 1$, as this results in an error value of approximately 1% [20].

Salmon and Warren [46] show that criterion 4.4 can cause large errors in some cases where the center of mass is near the edge of the cell. A modified opening criterion which circumvents this problem is

$$\theta < \frac{l}{D} + \delta, \tag{4.5}$$

where δ is the distance between the center of the mass and the geometric center of the cell. This addition ensures that if the center of mass is near the cell edge, it is only used for a force evaluation if removed by an extra distance of δ. If the center of mass is near the geometric center, it reverts to the previous criterion.

4.3.2 Complexity

By using the BH octree, we can compute a close approximation of the gravitational force in $O(N \lg N)$ time. There are two steps to the process, that of constructing the octree, and that of walking the tree to compute the forces.

Tree construction can proceed by iteratively inserting particles. Each particle insertion will walk down the tree, resulting in the addition of a new leaf node. As the height of the tree will scale as $O(\lg N)$, inserting all particles will scale as $O(N \lg N)$.

The force computation for a single particle will scale as $O(\lg N)$ for large N. This can be demonstrated by considering the example of a homogeneously distributed

system whose particle count is increased by a factor of eight. This is similar to adjoining seven root nodes onto an initial root node. The amount of additional force terms will increase by a constant amount dependent upon θ, not on the total number of particles in the system. Thus, while system size is increased by a factor of eight, force calculations per particle increase only by a constant factor. The scaling of a single force calculation is therefore $O(\lg N)$, and for the whole system $O(N \lg N)$.

"I need you to think of solutions to problems we haven't seen yet. I want you to try things that no one has ever tried because they're absolutely stupid."

Ender Wiggin

5

Numerical Methods

At this point, we have described the details regarding the numerical implementations of magnetohydrodynamics and gravity. In this chapter, we discuss numerical methods which are necessary to perform numerical simulation, but which are not intrinsic aspects of the MHD and gravity algorithms and are generally applicable. These will be the nearest neighbour search algorithm utilised for our implementations of SPH, the numerical integration techniques, and methods for controlling the size of the numerical integration time step. Also discussed will be the embedded scripting used to initialise new simulations.

5.1 Scripted Initialisation

Simulations are initialised using Lua scripts. Lua [23] is a scripting language which allows for strong integration with a host program. Using scripts provides the user with the abilities of a full programming language with which to use to define the initial state of the system. Many scientific softwares initialise simulations using an

inputted text file containing parameter values and switches, but this does not match the power and expressiveness that scripting provides.

Initial particle data (position, velocity, etc) is calculated in the Lua script, and then passed in to the main program. Scripts are also granted the ability to instantiate new systems, add operators, and specify the choice of integrator. Once the script is finished, execution of the main program begins.

Another type of functionality implemented is the ability for scripts to apply a set of operators to a partially initialised system, as well as to read data from that system. An example of the applicability of this functionality is the gas collapse test case of chapter 7. The initial conditions specify the internal energy of each particle. To instead initialise the system using entropy requires knowledge of particle densities, as the conversion between internal energy and entropy is a function of density. However, the densities are not known. To obtain the densities, the script applies the operators to calculate smoothing lengths, generate neighbour lists, and then compute the densities of each particle which are then read back in to the Lua script.

5.2 Nearest Neighbour Search

Neighbour finding in SPH simulations is an important optimisation. Each particle interacts only with a small subset, N_{neigh}, of the total set of particles, N, and much computational effort can be wasted on inefficient searching for that subset. The cell linked list method is popular in molecular dynamics because of its $O(N)$ scaling, however this efficiency is not preserved in astrophysical SPH simulations where interaction lengths can vary immensely over space. The uniform mesh, which is a cornerstone of the cell linked list method, is not suitable for these types of simulations.

Many of the astrophysical SPH codes developed recognize this problem and perform

neighbour searching through use of an space partitioning tree [49, 45, 54, 22]. This has two main benefits. First, the resolution of the tree is adapted to the particle distribution, creating a mesh that fits the density profile of the system. Second, in these codes the tree structure will already have been created during gravity routines, allowing the $O(N \lg N)$ computational effort spent in its construction to be considered as free work.

The method presented in this thesis constructs a new mesh from the nodes of the tree. In this mesh, all nodes are spatially exclusive, and each stores a local map of its physically adjacent neighbours. This provides the ability to hop laterally between tree branches without needing to traverse vertically through the tree. Thus, the search can discover outwards from the target, searching only those nodes which lie inside the search radius. An octree has been used for the choice of tree, but the principles are straightforwardly applicable to other space partitioning trees.

5.2.1 Linked List Cell Method

For the linked list cell method, the simulation space is overlayed with a rectangular mesh that bins all particles. For this mesh, the width of each cell is greater than or equal to the interaction length. By construction then, all neighbours of a particle are guaranteed to lie within the current and surrounding cells, as the cube this forms will completely enclose the sphere of interaction for all particles in the center cell. The remaining cells in the mesh need not be checked. For a homogeneous particle density (which, for molecular dynamics simulations, holds true on average), each particle will perform a constant amount of computational work to search these cells regardless of the total simulation size. This makes the total cost scale as $O(N)$.

Each cell in the mesh uses a linked list to store the particles it contains. This

Algorithm 1 Linked list cell construction

```
for each cell i in the mesh do
    Reset the link of i's head node
end for
for each particle i in N do
    Compute cell j which contains i
    Attach the list node of i to cell j's linked list
end for
```

allows for an efficient way to handle the issue of variable list sizes, as there is only a one time cost of allocating the memory for N list nodes (one for each particle), after which on each time step only the links of the nodes need to be updated. The process for list construction proceeds as given in algorithm 1 and is run each time step. It is possible to use a scheme which updates only those portions of the list which have altered between steps, but constructing the lists anew is simple and runs in $O(N)$ time, so potential savings are minimal.

Particle neighbours can found using the linked lists as shown in algorithm 2. The procedure is to iterate through all cells in the mesh, and for each cell, through their linked list. To avoid double checking particle pairs, only particles that are past the target particle in the list are checked, and only half of the surrounding cells are checked. For example, in two dimensions, only the four cells to the northeast, due north, northwest, and due west need to be checked. This cuts this eight cell ring around the target cell symmetrically in half, ensuring that all cell pairs are checked and no cell pair is checked twice. A similar arrangement is used for the three dimensional case.

5.2.2 Octree Neighbour Searching

However, the linked list cell method has particular problems with SPH that make it undesireable. Since the density of the fluid may vary greatly over space and each

Algorithm 2 Linked list cell method.

```
1: for each cell i in the mesh do
2:   for each particle j in i's linked list do
3:     Scan particles past j in i's linked list
4:     Scan all particles in the cells surrounding i
5:   end for
6: end for
```

particle has its own value for the smoothing length, the neighbour search may include too many particles for too coarse of a mesh, or have to search beyond the surrounding cells to find all neighbours. A constant cell size is not optimal for the whole simulation space.

Using a mesh whose resolution is adapted to the density topography of the system would alleviate the previous concerns. One such mesh is the octree, which has the benefit that it will have been already created during gravity routines.

The method of Hernquist and Katz [22] performs a tree walk of an octree to search for nodes that overlap with a cubical search radius, pruning paths of the tree that fall outside the search area. This has two main disadvantages. First, nodes that are outside the search region need to be explicitly checked and cut. Second, the height of the tree has to be traversed to access those nodes containing the set of neighbour particles. Thus, this search method will scale as $O(N_{\text{neigh}} + \lg N)$, that is, with the sum of the desired number of particle neighbours, plus all the paths (correct and incorrect) down the tree. What is desired then is a method that removes these problems, that is, which gives direct access to the nodes containing the neighbouring particles, and that does not need to explicitly cut out distant regions.

5.2.3 Local Map Searching

To alleviate the problems of tree walking searches, a new method for nearest neighbour searching was developed. Using the nodes of the octree, a new mesh is created that is connected in a manner that is logistically appropriate for nearest neighbour searches. In much the way a tree node retains information about its children, it now also retains information in a local map about physically adjacent nodes. As node dimensions can vary between neighbouring nodes, adjacent nodes are those that share a common boundary. A node may have multiple smaller nodes along one boundary, and they are all stored in its local map.

These nodes connect together to form a web of local maps, and this web can be generated alongside the construction of the octree itself. This is accomplished by applying three steps during a node split, when a parent node is subdivided into eight children nodes. The parent, which contains a local map of its neighbours, needs to push its local map downwards onto its children. The three steps are:

1. Make children appropriate neighbours of each other.

2. Remove parent from the local maps of its neighbours.

3. For each parents neighbour, add it to the local map of adjacent children, and add such children to its local map.

The result of these steps is that as the tree deepens, the parent node is replaced by its children in the web. This ensures spatial exclusion.

Given a particle and the node in the local map web which contains it, we can find its neighbours by performing a breadth first search (BFS) out through the local map web. This method puts priority on those nodes that have the shortest connection

distance from the target, expanding the frontier of the search volume in a uniform manner.

The BFS proceeds by adding search targets to a queue, which get marked to prevent their readdition. When a node is popped from the queue, the particles it contains are examined, and all unmarked nodes in its local map which overlap the search region are added into the queue. The search can be initialised by adding the node containing the target particle to the queue, and once the queue is exhausted, the neighbour list will be complete.

Using only the leaf nodes of the octree to form the local map web is undesirable because this then requires the inclusion of empty leaf nodes to prevent gaps from forming. Not only do such nodes then have to be stored, but since they do not contain any particles, searching them is only a waste of time. For three dimensional octrees, up to six of the eight nodes created during a split may be empty, and this adds a considerable amount of wasted nodes. To circumvent their inclusion in the web, whenever a node split will create an empty child, the propagation of the local map is halted at that node.

Of course, other tree choices, such as kd-trees, will inherently avoid this problem of empty leaf nodes. However, the meshes produced may still be too fine grain. In terms of floating point operations, the check to see if a node overlaps the search region is roughly equal to checking if a particle is inside that search region. Thus, the desire is to let nodes increase in size so that we search the same volume with fewer cells, while at the same time preventing them from growing too large to avoid searching too much excess volume, and correspondingly, too many particles.

This has been controlled by a parameter σ, which defines a population limit for nodes and regulates the coarsity of the local map web. During construction, nodes only propagate their local map while they are above this limit. Figure 5.1 shows

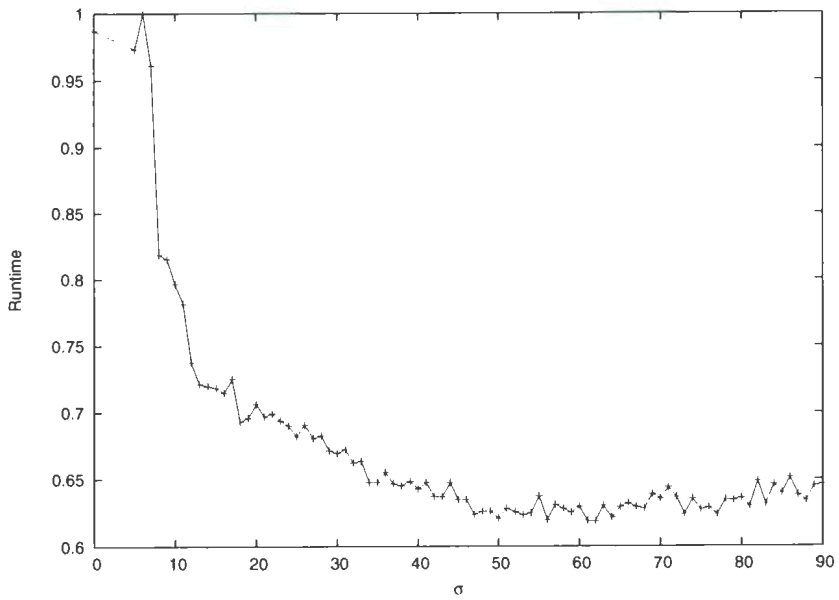

Figure 5.1: Efficiency gains for increasing σ with an ideal neighbour count of 60.

dramatic efficiency gains by increasing σ only even slightly, and an overall good choice for σ is about parity with the ideal neighbour count.

A list of the nodes composing the local map web is kept to allow for quick iteration without the need for a tree walk. This list can be created alongside the octree construction. A linked list is also used to track which node each particle belongs to. As in the cell linked list method, the size of this list can be fixed to N, since no particle needs to be doubly stored, and this allows for quick retrieval of particles within each node.

5.2.3.1 Adaptive Smoothing Length Update

It is understood that when the smoothing length is individually set per particle, it is best to minimize the variance of the number of particles enclosed by the smoothing length to improve the conservation of energy [35]. Additionally, recent SPH formalisms derived through a variational approach [50, 29] require the number of particle numbers

to be held exactly constant. For these reasons, the impetus exists to completely remove the variance of the number of neighbours, something which is not strictly guaranteed through current smoothing length estimation methods.

The BFS search method may be adjusted to accommodate this requirement, and set the smoothing length to encompass precisely the desired number of neighbours. Since the search moves radially outward from the target, we can locate the closest N_{neigh} particles, adaptively refining the smoothing length as the search progresses. This incurs some extra cost, as before we could blindly add all particles inside the smoothing length to the neighbour list. Now we are searching for exactly N_{neigh} particles, and each particle has to be compared against the others to see if it should be added to the neighbour list or not.

The procedure used is to first trivially accept nodes into the search queue, and particles into the neighbour list, until the list contains N_{neigh} particles. This gives an initial estimate for the smoothing length, at which point the search can start being refined. Future nodes are checked against the current smoothing length estimate as to whether they should be added to the search queue, and similarly for particles with the neighbour list. If a particle is found to be inside the smoothing length estimate, the most distant particle in the neighbour list is removed and the new particle is inserted to the list. The smoothing length estimate is then updated to be midway between the removed particle and the subsequent most distant particle in the neighbour list, which may not necessarily be the one just inserted. This requires an extra search through the neighbour list to find this most distant particle, which is the primary source of extra computational cost. Once the search queue is exhausted, the neighbour list will be populated with only the closest N_{neigh} particles, and the smoothing length will wrap only those particles.

5.2.3.2 Performance

In estimating the complexity of this algorithm, there are two aspects to consider. First is the creation of the local maps. This is embedded inside the construction of the tree by applying the three rules of local map generation at the time of a node split. This adds only a constant amount of work to this step, and the tree construction will retain its $O(N \lg N)$ nature.

The search portion of the algorithm revolves about the breadth first search, which scales with the sum of edges and vertices traversed in a graph. Here we can identify the number of edges traversed with the number of nodes examined. The vertices of the graph represent the number of searched items, which are the particles contained in the nodes. Since the search is constrained to nodes inside the smoothing length, the number of particles searched will correspondingly scale with the number of desired neighbours, N_{neigh}. In the worst case of one particle per node, the number of nodes visited will similarly scale with N_{neigh}. However, by using the optimisations detailed earlier to scale the population size of nodes in the local map web with N_{neigh}, the number of nodes visited can be reduced to a constant amount. Thus, the total search scales as $O(N_{\text{neigh}})$.

The BFS search method was compared against Hernquist and Katz's tree walk approach. The test case used was of a three dimensional spherical ball of gas which has a r^{-1} density profile. Both methods were run for the same particle positions and smoothing lengths. Figure 5.2 presents the ratio of search times, where there is a clearly observed efficiency gain, which grows for increasing particle counts. This is to be expected as the tree walking approach depends upon the size of the tree, while the BFS method does not. Of importance to note is that the additional time to create the local maps during octree construction has been added to the BFS values.

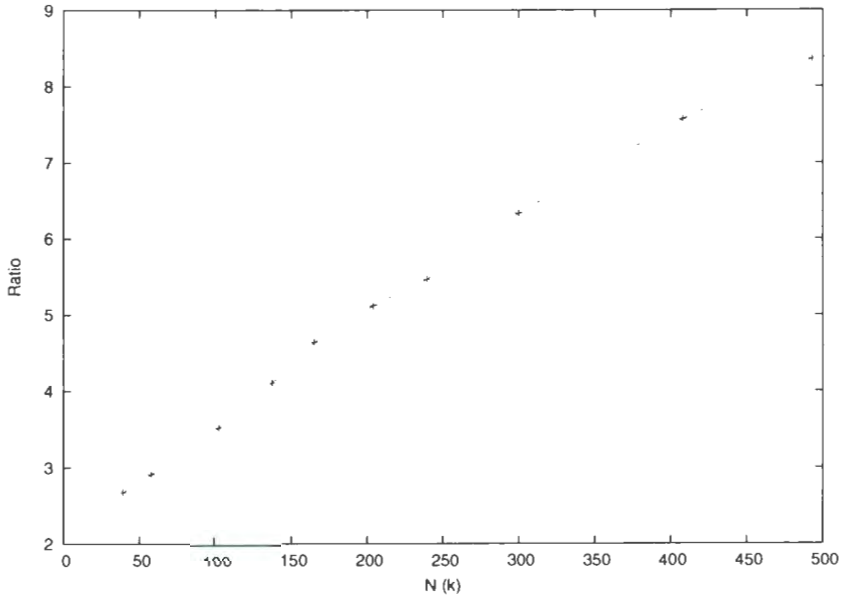

Figure 5.2: Ratio of search times between tree walking search and BFS.

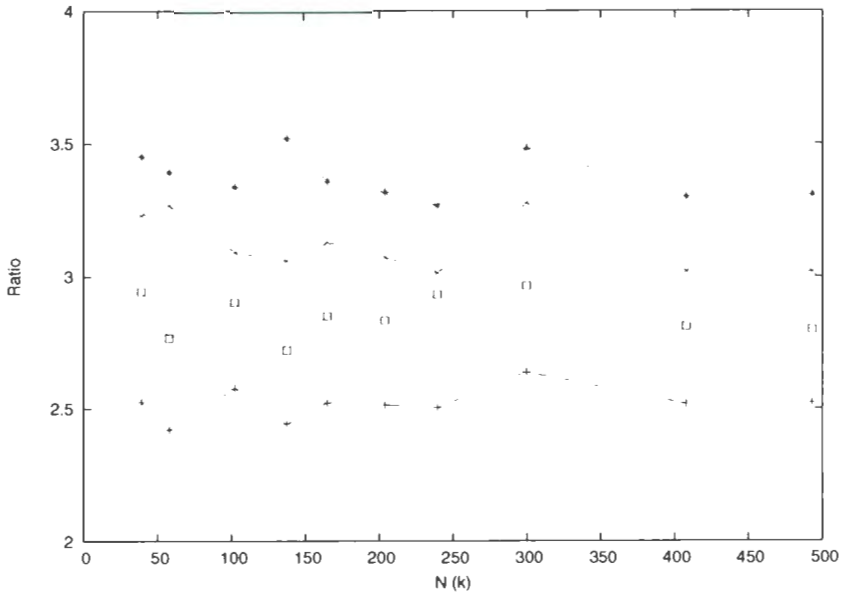

Figure 5.3: Runtimes of adaptive smoothing length updates, scaled in terms of equivalent number of searches, for the embedded BFS and iterative methods. The first, second, and fourth curves from the top represent the 10%, 5% and 1% iterative scenarios, with the third curve for the embedded BFS method.

The adaptive smoothing length update can be compared with Springel's iterative approach [49], since both will produce smoothing lengths containing exactly N_{neigh} particles. His approach performs repeated neighbour searches iteratively adjusting the smoothing length between steps until a smoothing length is found which contains the exact number of neighbours desired. The first step begins with an initial seed for the smoothing length chosen in some manner. This approach can be measured in terms of the number of searches performed per particle. In the best case, this will be a single search if the initial seed is perfect. The average or worst case will depend on the quality of the seed, and the number of steps required to reach convergence. What is important to consider is this variability in execution times. The embedded BFS approach performs only one search, but with additional overhead per searched particle. While it cannot achieve the best case scenario of the iterative approach, the runtime variability of smoothing length update step is removed.

An example is presented to demonstrate this effect. The same system as above is used, and the smoothing lengths are first updated with the embedded BFS method. These ideal values are then perturbed randomly within ranges of 1, 5, and 10 percent and the iterative approach is used to recalculate the smoothing length. Here we use the more efficient BFS method for the iterative method, not tree walking searches. From figure 5.3, the variance of the iterative method by altering the initial seeds can be seen. For seeds which are close to the ideal smoothing length, we approach the best case scenario and can outperform the embedded BFS method. As the seed grows further from the ideal smoothing length, the number of iterations required increase and performance drops.

5.3 Methods for the Numerical Integration of Ordinary Differential Equations

In this thesis, "numerical integration" refers to numerical estimation of solutions to sets of ordinary differential equations. In initial value problems, the integration procedure transforms an initial state to some future state. The role of the integrator is to follow the path determined by the physics as faithfully as possible. The choice of numerical integration technique has a significant impact on the speed, accuracy, and overall quality of simuation, and a poor or improper choice can lead to incorrect results.

The integration methods which were implemented in this work are detailed below. The methods are discussed in the context of solutions of equations of motion, where the symbol r^n represents the vector of positions, v^n the vector of velocities, and a^n the vector of accelerations for the system at step n. The size of the time step is denoted by Δt.

5.3.1 Euler

The simplest integration scheme is the Euler method. It calculates the derivative of a system variable, and then uses that to advance one step forward in time. For example, position and velocity variables are updated as

$$r^{n+1} = r^n + v^n \Delta t, \tag{5.1a}$$

$$v^{n+1} = v^n + a^n \Delta t. \tag{5.1b}$$

Alternatively, one can use the derivative at the end of the time step, given as

$$r^{n+1} = r^n + v^{n+1}\Delta t, \tag{5.2a}$$

$$v^{n+1} = v^n + a^{n+1}\Delta t. \tag{5.2b}$$

This is known as the implicit Euler method, and requires a root finding procedure to find the solution.

However, for partitioned systems, we can evolve one variable using the explicit Euler and the other by implicit Euler. For our position and velocity example, we can create a scheme of

$$r^{n+1} = r^n + v^{n+1}\Delta t, \tag{5.3a}$$

$$v^{n+1} = v^n + a^n\Delta t, \tag{5.3b}$$

which is known as the symplectic Euler method. Symplectic methods will be discussed in more detail in section 5.3.5. In this example, the implicit step can be computed exactly by first advancing the velocities, and then using them to advance the positions. Alternatively, the positions can be advanced first, this being feasible if the acceleration is a function of r only (as is usual for gravitational problems.)

Considering the Taylor series of a variable $y(t)$,

$$y^{n+1} = y^n + y'\Delta t + \frac{1}{2}y''(\Delta t)^2 + O\left((\Delta t)^3\right), \tag{5.4}$$

we can see that the leading truncation error in an individual Euler step will be proportional to $O\left((\Delta t)^2\right)$. Thus, this scheme is said to be first order accurate.

5.3.2 Leapfrog

Leapfrog integration, also known as Verlet integration, is a second order symplectic integrator. It is applied to partitioned systems, but advances the two equations out of phase with each other yielding a scheme of

$$r^{n+1/2} = r^{n-1/2} + v^n \Delta t, \tag{5.5a}$$

$$v^{n+1} = v^n + a^{n+1/2} \Delta t. \tag{5.5b}$$

Each variable advances using information half a step forward in time, but since the two equations are out of phase by half a step, this information is always explicitly available.

The above steps can be rewritten in the following equivalent form:

$$r^{n+1/2} = r^n + \frac{1}{2} v^n \Delta t, \tag{5.6a}$$

$$v^{n+1} = v^n + a^{n+1/2} \Delta t, \tag{5.6b}$$

$$r^{n+1} = r^{n+1/2} + \frac{1}{2} v^{n+1} \Delta t, \tag{5.6c}$$

so that both variables are evolved with matching initial and end time positions. It may also be seen more clearly from this that one leapfrog step is actually the composition of two symplectic Euler steps, where the integration of each variable alternates between implicit and explicit Euler steps.

5.3.3 Runge-Kutta

Runge-Kutta integration encompasses a family of iterative methods. The philosophy behind such methods is to produce derivative values at different points across a time step, and then to complete a step forward using a weighted average of those points. Second and fourth order Runge-Kutta methods have been implemented in this work using the common weighting values, and will be discussed. It is worth noting that the Euler method can be considered a first order Runge-Kutta method.

The trapezoidal rule is used for the second order Runge-Kutta method (RK2), which is given by the scheme

$$k_1 = f\left(y^n\right), \tag{5.7a}$$

$$k_2 = f\left(y^n + k_1 \Delta t\right), \tag{5.7b}$$

$$y^{n+1} = y^n + \frac{1}{2}\left(k_1 + k_2\right)\Delta t \tag{5.7c}$$

which can represented more conveniently using a tableau format as

$$\begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \tag{5.8}$$

The fourth order Runge-Kutta method (RK4), also known as "The" Runge-Kutta method because of its popularity, has a scheme of

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & \frac{1}{2} & \\ \hline & \frac{1}{6} & \frac{2}{6} & \frac{2}{6} & \frac{1}{6} \end{array} \tag{5.9}$$

These methods will have higher accuracy than the Euler method, but at an increased computational cost. Two derivative evaluations are required per time step for RK2, and RK4 requires four derivative evaluations. The accuracy gained from using a higher order method, however, should allow for the use of larger time steps, thereby reducing the impact of these increased computational costs.

5.3.4 Runge-Kutta-Fehlberg

Runge-Kutta-Fehlberg integrators falls into a class of techniques known as embedded Runge-Kutta methods. The premise of such methods is to perform two Runge-Kutta integrations of seperate orders per step, but to choose them such that some of the intermediate steps of the lower order method coincide with steps of the higher order method. The higher order method is used to generate a measure of error of the lower order step, and by using an embedded Runge-Kutta method, the computational effort necessary to compute this error can be reduced.

Such a procedure was first developed by Fehlberg, and is commonly known as Runge-Kutta-Fehlberg (RKF) integration. The method implemented here is the "common" fourth order scheme [15], which is of a fourth order Runge-Kutta embedded in a fifth order, and is known as RKF4(5).

The scheme,

$$
\begin{array}{c|cccccc}
0 & & & & & & \\
\frac{1}{4} & \frac{1}{4} & & & & & \\
\frac{3}{8} & \frac{3}{32} & \frac{9}{32} & & & & \\
\frac{12}{13} & \frac{1932}{2197} & -\frac{7200}{2197} & \frac{7296}{2197} & & & \\
1 & \frac{439}{216} & -8 & \frac{3680}{513} & -\frac{845}{4104} & & \\
\frac{1}{2} & -\frac{8}{27} & 2 & -\frac{3544}{2565} & \frac{1859}{4104} & -\frac{11}{40} & \\
\hline
 & \frac{25}{216} & 0 & \frac{1408}{2565} & \frac{2197}{4104} & -\frac{1}{5} & 0 \\
 & \frac{16}{135} & 0 & \frac{6656}{12825} & \frac{28561}{56430} & -\frac{9}{50} & \frac{2}{55}
\end{array}
\tag{5.10}
$$

evaluates six derivatives to produce a fourth order step, y^{n+1}, and a fifth order step, z^{n+1}.

We can then use the difference, $\delta = |z^{n+1} - y^{n+1}|$, to compute the leading truncation error of the lower order method. Based upon this error measure, we can either accept the lower order step as being accurate within some tolerance, or reject it for being too inaccurate. If it is rejected, y^n is not advanced, and instead this integration step is repeated for a smaller step size. Otherwise, if it is accepted, y is advanced to y^{n+1}. It is tempting to use z^{n+1} instead, since it is a result from a higher order method, but it is the error of the lower order scheme which is known. The error associated with the higher order scheme is uncertain, and therefore should not be used. Details regarding step size control using the error estimate are given in section 5.4.3.

5.3.5 Symplectic Integration

For Hamiltonian systems, a canonical transformation is one which preserves the area of phase space. Owing to this, phase space can be considered as a symplectic manifold,

since though its shape may change over time, its area remains constant. A numerical integration method is then considered symplectic if it transforms a Hamiltonian system from time t^n to t^{n+1} as a canonical transformation.

This property of a numerical integrator will ensure that the system remains Hamiltonian, and provides exceptional long term stability. An example to highlight the power of this property is presented. It consists of a six body system representative of the solar system, where the Sun has the mass of the inner terrestial planets added to it, and the outer planets are individually modeled (including Pluto.) The reference frame is fixed to the Sun so that it remains at rest. The initial conditions correspond to September 5, 1994 at 0h00 [19] and are given in Table 5.1. The system is described using distances of AU, time in Earth days, and with the gravitational constant of

$$G = 2.95912208286 \cdot 10^{-4}.$$

The system is evolved seperately using the symplectic leapfrog and the non-symplectic RK4 integrators using a fixed step size of 200 Earth days. Figure 5.4 shows the resulting evolution. While initially the orbits of Jupiter are more accurate using the higher order RK4 integrator, its long term behaviour fails. The cumulative effects of the non-symplectic time integration leads to an energy leak, causing Jupiter to spiral inwards towards the Sun and eventually be ejected.

5.4 Time Step Control

The size of the integration time step is of critical importance to the accuracy and stability of a simulation. An infinitesimal time step would be most representative of the continuous flow of time, but this is not possible. Instead, we take discrete, finite

Body	Mass	Position	Velocity
Sun	1.00000597682	0.0	0.0
		0.0	0.0
		0.0	0.0
Jupiter	0.000954786104043	-3.5023653	0.00565429
		-3.8169847	-0.00412490
		-1.5507963	-0.00190589
Saturn	0.000285583733151	9.0755314	0.00168318
		-3.0458353	0.00483525
		-1.6483708	0.00192462
Uranus	0.0000437273164546	8.3101420	0.00354178
		-16.2901086	0.00137102
		-7.2521278	0.00055029
Neptune	0.0000517759138449	11.4707666	0.00288930
		-25.7294829	0.00114527
		-10.8169456	0.00039677
Pluto	$(1.3 \cdot 10^8)^{-1}$	-15.5387357	0.00276725
		-25.2225594	-0.00170702
		-3.1902382	-0.00136504

Table 5.1: Masses and initial positions and velocities for the outer solar system. The mass of the Sun has the terrestial planets added into it. Quantities are given in solar mass, AU, and AU per Earth day, respectively.

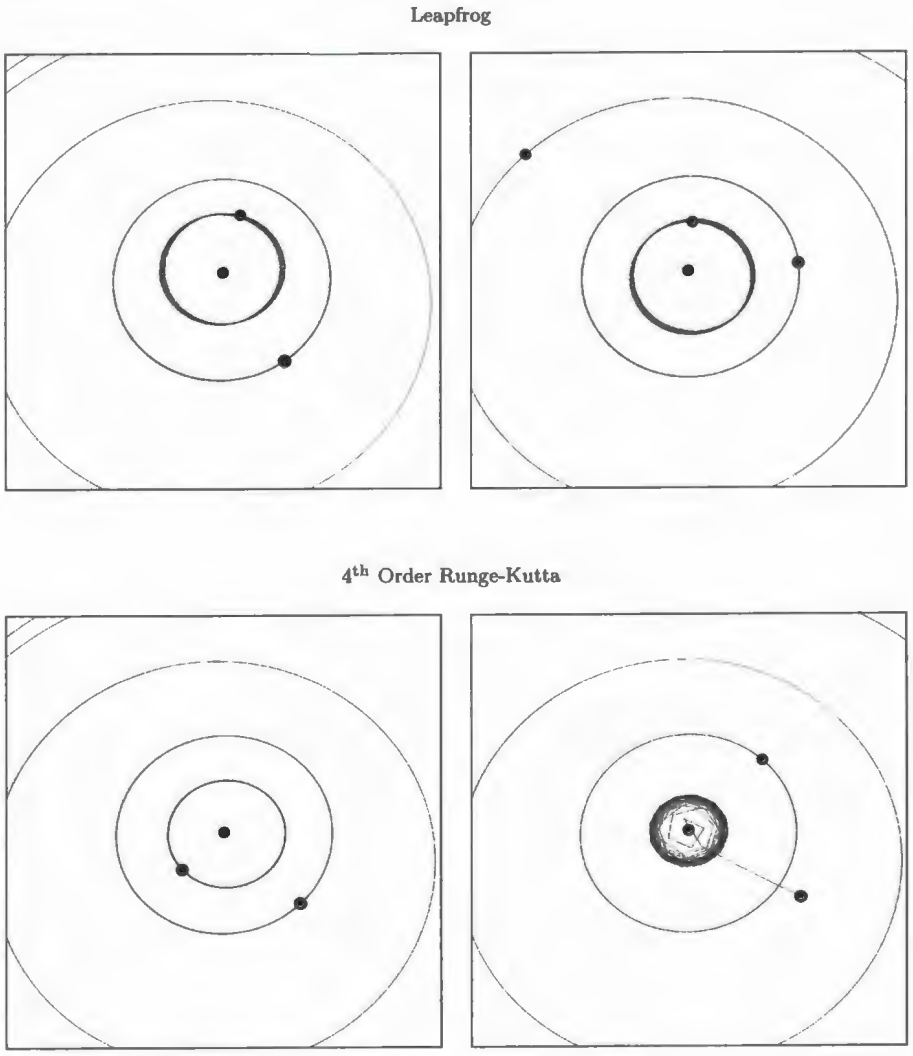

Figure 5.4: The outer solar system integrated using leapfrog (top) and RK4 (bottom). The time step is fixed at 200 earth days. The lefthand figures show the interval $0 \leq t \leq 100$ years, and the righthand figures show the interval $3365 \leq t \leq 3465$ years.

sized steps, the size of which should be chosen to balance the competing desires for short steps to improve accuracy, and longer steps to reduce computational workload. The primary question can be stated as thus: how large of a step can be taken and still faithfully simulate the physics?

A simulation may gain incredible efficiency by asking this question of every integration step. This is known as adaptive time stepping, where the size of the time step is chosen anew each step. The variety of methods available to make this choice will be discussed.

5.4.1 Particle Kinematics

The first approach uses information from the current state of the system to estimate a time step size. The size is calculated using the kinematics of the SPH particles in order to restrict their position change as a function of their smoothing lengths. By choosing the time step in this way, the configuration of the particles is prevented from being altered too greatly in any single step.

The relations used to calculate the time step size are derived from simple kinematic equations. The new step value is defined by

$$t_{\text{new}} = \min\left[\frac{Ch_a}{v_a}, \sqrt{\frac{Ch_a}{2f_a}}\right], \quad a \in \{N\} \tag{5.11}$$

where f refers to the net force. A Courant like hydrodynamic constant, C, is used to govern the degree of position change, with common values between 0.3 - 0.5.

There exists a crucial flaw to this time stepping scheme, however. If we consider the example depicted in figure 5.5 of a particle A which has information to communicate to particle C, we see that A cannot directly communicate this information since each particle can only interact with particles inside its smoothing length. Thus, A

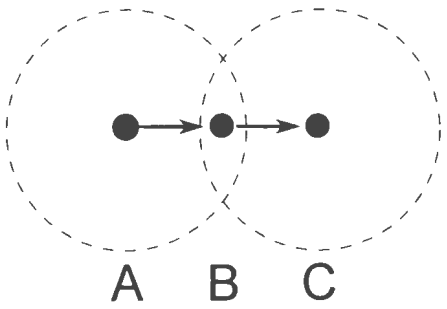

Figure 5.5: Information from A to C has to travel through B.

must first communicate to B, who in turn communicates to C. If the physics of the system dictate that the information should be relayed in real time t, then the net time for these two steps should not exceed this amount otherwise this event will fail to be modeled properly. However, the kinematic time stepping scheme provides no measure of this situations veracity.

5.4.2 Signal Velocity

The solution to the problem presented in the preceeding section is straightforward. A similar calculation can be performed using the speed of the fastest information wave to dictate step size instead of the kinematics of the particles. Between two moving observers this quantity is known as the signal velocity, of which a good estimate for it in hydrodynamic systems is

$$v_{ab}^{\text{sig}} = c_a + c_b - \beta \mathbf{v}_{ab} \cdot \hat{\mathbf{r}}_{ab}, \tag{5.12}$$

where β is one of the viscosity parameters, and $c_{\{a,b\}}$ is the speed of sound. This is similarly restricted to travel within a fraction of the smoothing length, with typical values of $C \sim 0.15 - 0.20$. The step size is defined then as

$$t_{\text{new}} = \max \left[\frac{Ch_a}{v_{ab}^{\text{sig}}} \right]. \tag{5.13}$$

In magnetohydrodynamic systems where there exists a variety of other possible information waves, the definition of the signal velocity is adjusted to use the speed of the fastest magnetic wave instead of the speed of sound. This is given by

$$v_a^{\text{sig}} = \frac{1}{\sqrt{2}} \left[\left(c_a^2 + \frac{B_a^2}{\mu_0 \rho_a} \right) + \sqrt{\left(c_a^2 + \frac{B_a^2}{\mu_0 \rho_a} \right)^2 - 4 \frac{c_a^2 \left(\mathbf{B}_a \cdot \hat{\mathbf{r}}_{ab} \right)^2}{\mu_0 \rho_a}} \right]^{1/2}, \tag{5.14}$$

with the signal velocity becoming

$$v_{ab}^{\text{sig}} = v_a + v_b - \beta \mathbf{v}_{ab} \cdot \hat{\mathbf{r}}_{ab}. \tag{5.15}$$

In MHD systems, values between 0.075 - 0.1 are typically used for the Courant like factor, half that of pure hydrodynamic systems.

5.4.3 RKF45 Time Stepping

In section 5.3.4, the fourth order Runge-Kutta-Fehlberg (RKF45) integrator was introduced. By its design, each integration step produces an estimate of the error of that step, which is used as part of an acceptance criterion and to calculate the size of the next time step. A constant value is used for an acceptable level of error per step, called the tolerance, and is used as a comparison against the error to accomplish these two tasks.

The difference between the two estimated solutions, δ, is used as the measure of error. It is important to note that δ is actually a vector of differences, since not only

may the integrated variable be a vector quantity (such as position and velocity), but there may be multiple integrated quantities. The commonly used approach is to take the Euclidean norm of δ and use the resultant value for the time step adjustment. We define

$$\text{err} = \sqrt{\frac{1}{n}\sum_{i}^{n}\left(\frac{\delta_i}{\text{tol}_i}\right)^2}, \tag{5.16}$$

with the sum being over the vector components of δ [39]. An individual tolerance, tol_i, may be specified for each component.

Using err, the new time step size is calculated as

$$\Delta t_{\text{new}} = s\Delta t_{\text{old}}\left|\frac{1}{\text{err}}\right|^{1/p}. \tag{5.17}$$

The parameter p customarily has its value chosen to equal the order of the integrator used. A safety factor s, a few percent smaller than unity, is included to help guard against overestimations of the step size.

5.4.4 Step Halving

The step halving approach, also commonly referred to as step doubling, is similar in principle to the RKF45 method. An error estimate is calculated each step which is used to adjust the time step, but this method is able to be applied to a wide range of integration techniques. The step halving method operates by repeating each integration step with two steps of half size. The two short steps provide a higher precision solution which can contrasted with the long step to obtain an error estimate. As in the RKF45 method, this error is then used in an acceptance criterion and to adjust the size of the next time step.

The implemented time step adjustment procedure is a modified version of the scheme used for the RKF45 method. After a successful step, the time step is increased with the RKF45 approach, but after a rejected step, it is decreased instead by a factor of two. This contains an inherent advantage in efficiency, as the previous first half step can be reused as the next long step.

5.4.5 Constant Step Size

While adaptive time stepping has many benefits, it is worthwhile to investigate some of the secondary effects it has on symplectic integration. Adjusting the time step is equivalent to a reparameterisation of the time variable in a Hamiltonian system from $t \to \tau$. This changes the differentiation of a system variable y from

$$\frac{dy}{dt} = f(y) \tag{5.18}$$

to

$$\frac{dy}{d\tau} = f(y)\frac{dt}{d\tau} \tag{5.19}$$

which follows simply from the chain rule. The desire then is to reparameterise t so that $f(y)\frac{dt}{d\tau}$ will retain the same geometric features as $f(y)$.

The simplest solution to achieve this is to hold the time step size constant, that is, to not reparameterise t. This trivially guarantees that the integration will remain symplectic.

An example to illustrate the danger of improper adaptive step sizing is presented, based upon an example from Hairer [19]. We consider the perturbed Kepler problem with dynamics

$$\begin{aligned} \dot{r}_x &= v_x \qquad \dot{v}_x = -\frac{r_x}{\left(r_x^2 + r_y^2\right)^{3/2}} - \frac{\delta r_x}{\left(r_x^2 + r_y^2\right)^{5/2}} \\ \dot{r}_y &= v_y \qquad \dot{v}_y = -\frac{r_y}{\left(r_x^2 + r_y^2\right)^{3/2}} - \frac{\delta r_y}{\left(r_x^2 + r_y^2\right)^{5/2}} \end{aligned} \tag{5.20}$$

where $\delta = 0.015$ and the initial state is given as $v = [0, \sqrt{(1+e)/(1-e)}]$ and $r = [1-e, 0]$. The problem considered has eccentricity $e = 0.6$. The system is advanced using the symplectic leapfrog independently with both a fixed step size 0.065, and using the step halving scheme with a tolerance of 0.002.

The two evolutions are presented in figure 5.6 along with the exact solution. It can be seen that although the adaptive step size implementation initially yields accurate results, the effects of the repeated non-canonical time transformations significantly affect its long term evolution. The defining characteristic of the system, that is of the elliptical orbit, is lost as the orbit decays towards circular. On the other hand, though the orbit from the fixed step implementation is different from the exact solution from the outset, it retains its eccentricity over long durations. For this reason, using a constant step size should still be given consideration when choosing a time stepping scheme.

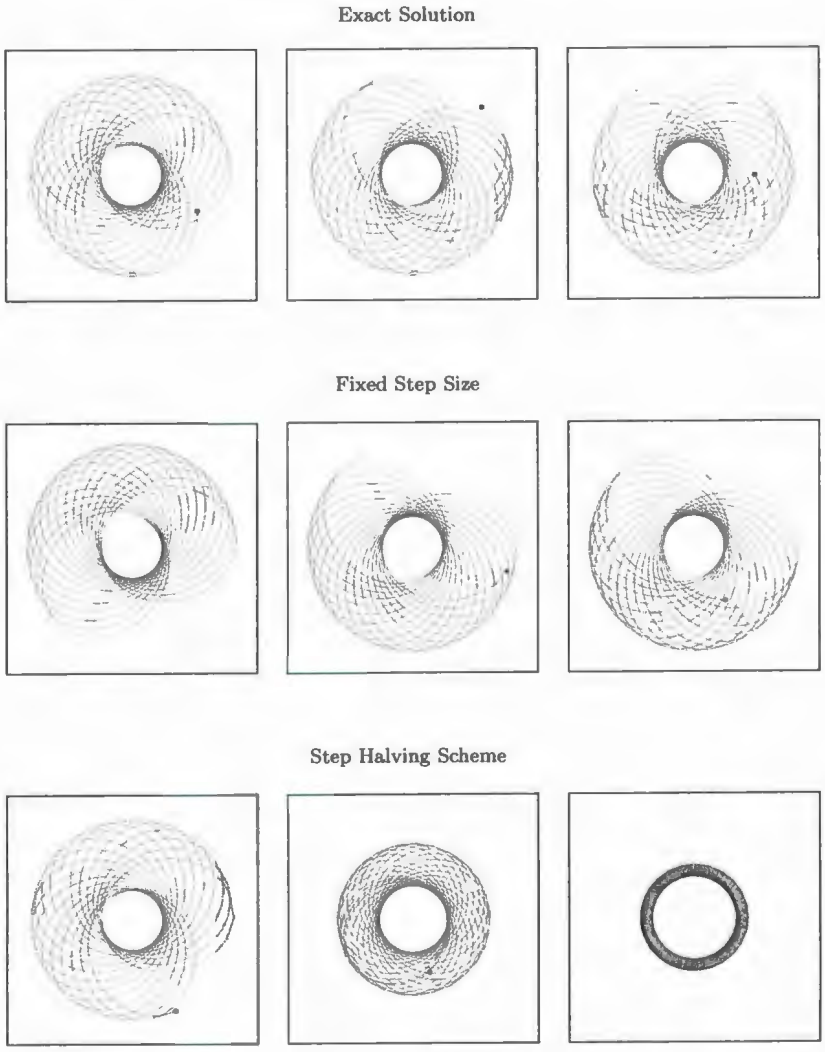

Figure 5.6: Evolution of the perturbed Kepler problem computed using leapfrog integration with fixed time step size (mid) and adaptive step halving time step control (bottom) compared with the exact solution (top). Lefthand figures trace orbits for $0 \leq t \leq 100$, center figures for $2400 \leq t \leq 2500$, and righthand figures for $4900 \leq t \leq 5000$.

"What men are poets who can speak of Jupiter if he were a man, but if he is an immense spinning sphere of methane and ammonia must be silent?"

Richard Feynman

6

Visualisation

A software suite was developed to analyse and interpret data generated from the SPH physics engine. Its design goal was to provide a simple way of performing the types of analysis commonly needed. There are two principle components: one plots system variables such as energy and momentum, and the second provides visualisation of the matter and fields which comprise the system.

The GUI (Graphical User Interface) elements of this program were built using Qt [16], which is a cross platform application and UI framework. The plotting program gnuplot [56] is used to handle the generation of plots, with this software functioning as a GUI frontend for it. This removes the direct handling and low-level manipulation of data away from the user, who can then focus on the essentials of data representation.

System visualisation is displayed using OpenGL. The user is given free range with the camera so that he is not locked into fixed viewpoints, and still images may be captured from the OpenGL window. Several types of visualisation techniques have been implemented, which are described below.

6.1 Particle View

The simplest way to visualise SPH simulations is with a particle representation, in which each particle is rendered as a simple sphere in the OpenGL scene. Figure 6.1 shows an example of this view. This common method provides a quick first check of system behaviour, but it is nevertheless a crude approach. Since the SPH particles are smoothed out over their local volume and overlap each other, it is more appropiate to represent this system as a continuum than discrete particles.

In this view, trajectory or "comet" tails may be turned on for each particle. This highlights the paths of particles over previous steps, with the number of steps being a user tunable value.

6.2 Kernel Imaging

Since the SPH particles represent a fluid, an approach that lets us visualise them as a fluid is of great value. One of the approaches used to accomplish this is the kernel imaging technique first used in the SPH visualisation program SPLASH [40].

A bitmap can be considered as a regular grid of pixels. If that is overlayed onto the system, each pixel will map to a specific point in the system. Using SPH interpolation, we can calculate values at each of these pixel sites, which in turn are used to define the colour of that pixel.

The procedure followed is to iterate over the particles, adding the contribution from each particle to the pixel sites within its smoothing length. A square area about the particle with width of the smoothing length is defined, and the pixel sites inside can be located quickly because of their regularly distributed nature. Figure 6.2 shows an example of the kernel imaging technique.

Figure 6.1: Particle view.

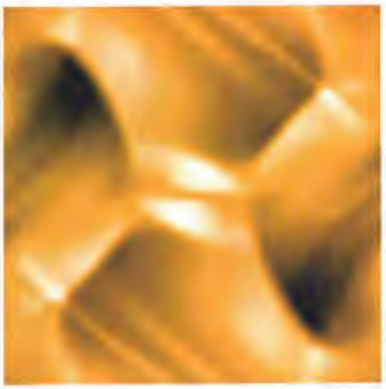

Figure 6.2: Kernel imaging.

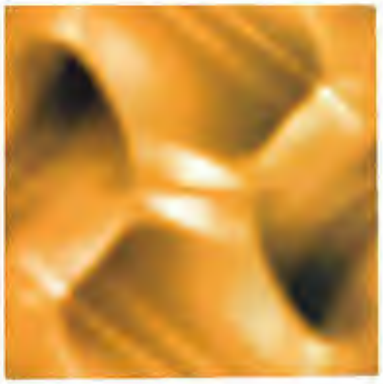

Figure 6.3: Triangulated image.

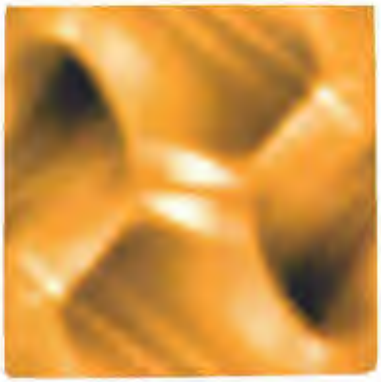

Figure 6.4: Triangulated image using only ten percent of the vertices.

The runtime for this method scales predominantly with the number of pixels in the image. If the pixel count doubles, each particle will then interact with twice as many pixel sites, and therefore the runtime has linear dependence on pixel count. However, as particle count increases, typically the smoothing lengths of those particles decrease, and each particle will subsequently interact with fewer pixel sites. If the smoothing lengths of particles are adjusted to keep the number of neighbours roughly constant, then we can consider the pixel sites have a "pseudo"-smoothing length for which this holds true as well. Thus, while there is some increased overhead by increasing particle count, the runtime is primarily driven by the pixel count of the generated image.

If the pixel count of the generated image is too small in relation to the particle count, features of the system could fail to be represented. In the limiting case of a one pixel image, there clearly will be a significant loss of information. The threshold at which this loss of information begins to occur is when the inter-pixel distance is greater than twice the smoothing length, as particles then can be "lost" inside the cracks between pixel sites. In practice, this situation is rarely a concern, as even for a small image of 100^2 pixels, it requires a particle count $\sim 10^6$ before such worry is necessary.

6.3 Triangulated Imaging

A second method for visualising the SPH fluid as a continuum is to generate a surface using a triangulation technique, with the particles serving as the triangle vertices. This surface can then be coloured using the particle data directly. The triangulation technique chosen is Delaunay triangulation, and the implementation has been adapted from Numerical Recipes [39].

Their code has been extended to include the special case of points lying on the same line, which they avoided with the reasoning that such occurences would be rare. The proper handling for such cases has been added, because for SPH, such occurences are common. Initial positions are often arranged on a lattice, and evolved states tend to form lattice-like structures.

The Delaunay triangulation algorithm used is an incremental one where vertices are added one at a time to the triangulation. A heirarchical storage scheme is used, and this gives an average runtime scaling as $O(V \lg V)$ [47, 38], where V is the number of vertices. In the worst case of a heavily lopsided tree, this will scale as $O(V^2)$, but can be (almost) guaranteed to be avoided by using randomised insertion of vertices

Algorithm 3 Uniform Particle Skipping

Require: V < N **and** npts > 0
1: ratio ← N / V
2: **for** j ← N **to** 1 **do**
3: **if** j / V < ratio **then**
4: vertices[V] ← j - 1
5: **end if**
6: V ← V - 1
7: **end for**

to keep the tree evenly distributed.

The resolution of the triangulated image can be adjusted by limiting the number of particles to use as triangle vertices. The difficulty is how to decide the subset of particles to use. Ideally, this set should be representative of the system with the coarsity of resolution being equal in all areas. A randomised selection works well, but a different approach has been implemented.

This approach selects a uniform sampling of particles from the initial system state, and uses this set for all future states. The initial particle arrangements are on a lattice, so choosing vertices in this manner is near optimal. As the particles follow the streamlines of the system, the quality of this selection remains high as the fluid evolves. Figure 6.3 shows an image generated with the triangulated image technique, and figure 6.4 is a comparison image with only 10% the number of vertices.

The uniform skip selection algorithm is detailed in algorithm 3. It utilises the ratio of N to V to determine when a particle is added to the vertex set. As all the particles in set N are iterated over, a counter initially set to N is decremented. A second counter initially set to V is decremented when a particle is added to the set of vertices. The decision of when to declare a particle as a vertex is made when the ratio of these counters falls below the target ratio. This procedure will select vertices uniformly across the set of particles, and works for any vertex count.

6.3.1 Delaunay Triangulation

A triangulation aims to subdivide an area into a set of contiguous and non-overlapping triangles, of which the area is defined by a set of points. These points act as the triangle vertices. Delaunay triangulation refers to a particular method of triangulation created by Boris Nikolaevich Delone in 1934 [39]. It is widely used still today because of its simplicity, but often more importantly, because it strives toward creating equiangular triangles. These are well balanced triangles that avoid pathological cases, such as the "long and skinny" type triangles generated by large angles.

The key design to accomplish this is through the use of circumcircles. Take for example the case in figure 6.5. Triangle ABC's circumcircle encloses the point D. This is considered "bad", and ABC would be undesireable. To remedy this, we perform an edge flip, flipping edge AC, creating edge BD and triangles ABD and BCD. The result, shown in figure 6.6, is that the circumcircles of both new triangles are now empty.

If all triangles in the triangulation satisfy this property of empty circumcircles, it is called a Delaunay triangulation. For any quadrilateral, one of the diagonals is guaranteed to yield triangles that have empty circumcircles. As such, a Delaunay triangulation is always possible to be generated. This diagonal bisects the largest angle of the quadrilateral. Doing so maximizes the minimum angle of the two triangles (bisecting a smaller angle would create even smaller angles), and this implies that triangles created will be as close to equiangular as possible.

Each edge in a Delaunay triangulation also have a similar circumcircle property. Any edge can be enclosed by a circle such that no other points lie inside it, and such edges are called Delaunay edges. All the edges of a Delaunay triangle are Delaunay edges, and a triangle formed from three Delaunay edges is a Delaunay triangle.

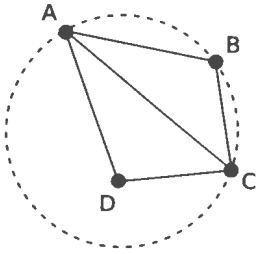

Figure 6.5: Example of circumcircles. Point D lies inside the circumcircle of triangle ABC.

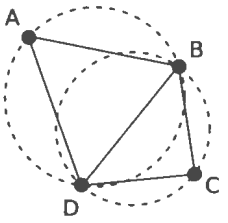

Figure 6.6: After the edge flip of AC. Both triangles circumcircles are now empty.

The algorithm used is an incremental one, where points are individually added to an existing triangulation, subdividing a triangle to accomodate the new point. As shown in figure 6.7, when a new point E is added to triangle ABC, it is subdivided into triangles EAB, EBC, and ECA.

We then check each of the three new triangles to see if any edges need to be flipped. The three inner edges are guaranteed to be valid. This can be seen by considering the circumcircle for ABC. We know that it was previously empty before the addition of E, and if it is shrunk in size to wrap around any of the inner edges, it will continue to remain empty. This proves they are Delaunay edges, and thus, the only edges that need to be checked are the three outer edges.

We must also consider the case of an inserted point lying on the edge of an existing triangle. In this case, we break this edge into two, and subdivide each triangle sharing that edge into two as shown in figure 6.8. Similarly, we only need to check the four outer edges for validity.

Let us examine when it is necessary to perform edge flips after the insertion of E. If we consider edge AB in triangle EAB (for either case), it is a common edge with another triangle GBA for some exterior point G. We know that GBA was a Delaunay triangle before the insertion of E, and so if AB does not need to be flipped, than

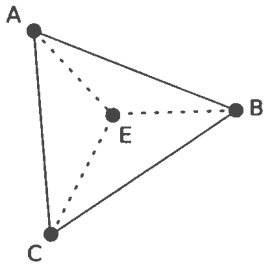

Figure 6.7: Addition of point E subdivides the parent triangle ABC into three triangles ABE, BCE, and CAE.

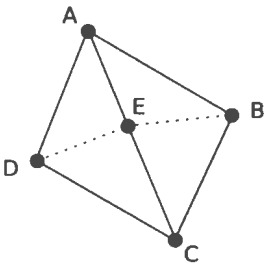

Figure 6.8: Insertion of a point onto the existing edge AC. Triangles ABC and ACD are subdivided to create triangles EAB, EBC, ECD, and EDA.

both EAB and GBA are Delaunay triangles, all their edges are valid and the full triangulation is Delaunay. If however AB is flipped, we still know that EA, BE and the newly formed EG are Delaunay edges, though this now puts the validity of edges AG and GB into question. While EAG and GBE are now locally Delaunay triangles when viewing just these pair of triangles in isolation, it is uncertain if they are globally Delaunay triangles. We need to examine them paired with the adjacent triangle along each questionable edge. Here, the situation is the same as the previous, and we can repeat this process until all edges are valid. The framework of the algorithm is given in algorithm 4.

An important detail to consider is how to handle the insertion of point that does not lie inside any triangle. A similar conundrum is how to perform the insertion of the first point. These situations can be averted by ensuring that points are always inserted inside a larger existing triangle. This is achieved by creating a huge "fictitous" triangle bounding all points to be inserted. After all point insertions are completed, any triangles that share a vertex with this triangle are discarded. This leaves the desired triangulation, which is still Delaunay since all edges remain Delaunay edges.

A tree data structure is used to store the triangles. When a triangle is subdivided,

Algorithm 4 Delaunay Triangulation

```
for each point R do
    Locate the triangle A containing R
    if R lies inside A then
        Subdivide A creating three child triangles
        Delete the parent triangle A
    else if R lines on the edge between triangles A and B then
        Subdivide A and B into two child triangles
        Delete both parent triangles A and B
    end if
    Recursively check validity of all outer edges
end for
```

its child triangles are added as leaves of that node, and in the case of edge flips, both new triangles are added as children to both parent triangles. Since each node contains at most three children, searching for the triangle containing a point to be inserted can be done in $O(h)$ time, where h is the height of the tree. [12, 11]

Finding adjacent triangles when performing edge flips can be done in $O(1)$ using hash tables. There are two tables used. One uses the three points of a triangle to find its index in the above tree data structure, and the other finds the third point of a triangle when given the opposite edge. For the edge hash, a complication arises because each edge belongs to two triangles, and therefore has two points opposite it. This is handled by always referencing triangles in clockwise order. Thus, edges common to two triangles can be differentiated by the order of their reference. For example, in figure 6.8, triangle ABC references the shared edge with ACD as CA, while ACD references it as AC. We can use this terminology to store a reference to point B using key $(h(C) - h(A))$, and the negation of this key, $(h(A) - h(C))$, to reference point D. Here h is a hash function.

The keys used in the triangle hash are stored as $(h(A) \oplus h(B) \oplus h(C))$ where $\oplus$ is the XOR operation. Therefore this key will have the same value regardless of the

ordering of the vertices.

"A process cannot be understood by stopping it. Understanding must move with the flow of the process, must join it and flow with it."

The First Law of Mentat

7

Test Cases

Several common test problems were used to verify the correctness of the code. There were three primary aspects that were examined: gravitational force calculations, hydrodynamic simulation, and magnetohydrodynamic simulation.

7.1 Gravity and Hydrodynamics

7.1.1 Solar System

A simple test performed of the gravitational routines was to model our solar system. This is the same system first presented in section 5.3.5. The leapfrog integrator with fixed step size 0.01 Earth days was used to conduct the evolution of the system.

The sidereal orbital period of each body from simulation, along with the expected real world values, is presented in table 7.1. We find close agreement between the two values, providing initial confirmation to the gravitational routines validity. The largest deviations exhibited are for the orbit of smallest radius (Jupiter), and orbit of

Body	Simulation	Real	Difference
Jupiter	4334	4332.589	0.033%
Saturn	10761	10759.22	0.017%
Uranus	30681	30685.4	0.014%
Neptune	60198	60189	0.015%
Pluto	90574	90465	0.121%

Table 7.1: Sidereal orbital periods for the solar system gravitational test. Values are given in Earth days. Real world values obtained from the NASA planetary fact sheet [55].

largest eccentricity (Pluto), which conform with expectations of errors arising from integration.

7.1.2 Spherical Gas Cloud Collapse

The primary test problem for the hydrodynamic routines was the adiabatic collapse of a spherical gas cloud, first presented by Evrard [14]. It served not only as verification of the hydrodynamic routines, but also further verification of the gravitational routines. As the gas cloud initially contracts under the influence of its gravity, its temperature and pressure will increase. Once the outward directed force of the pressure surpasses the inward directed force of gravity, a shock wave is sent from the interior to the outer regions of the system.

The gas cloud is spherically symmetric with an initial density profile of

$$\rho(r) = \frac{M}{2\pi R^2}\frac{1}{r} \tag{7.1}$$

where M is the total mass of the cloud and R is the cloud's radius. The units are chosen such that $M = R = G = 1$. The internal energy per unit mass is set to $u = 0.05\frac{GM}{R}$ with the specific heat ratio at 5/3. All particles are initially at rest.

The initial positions of the particles are distributed using Evrard's radial stretch

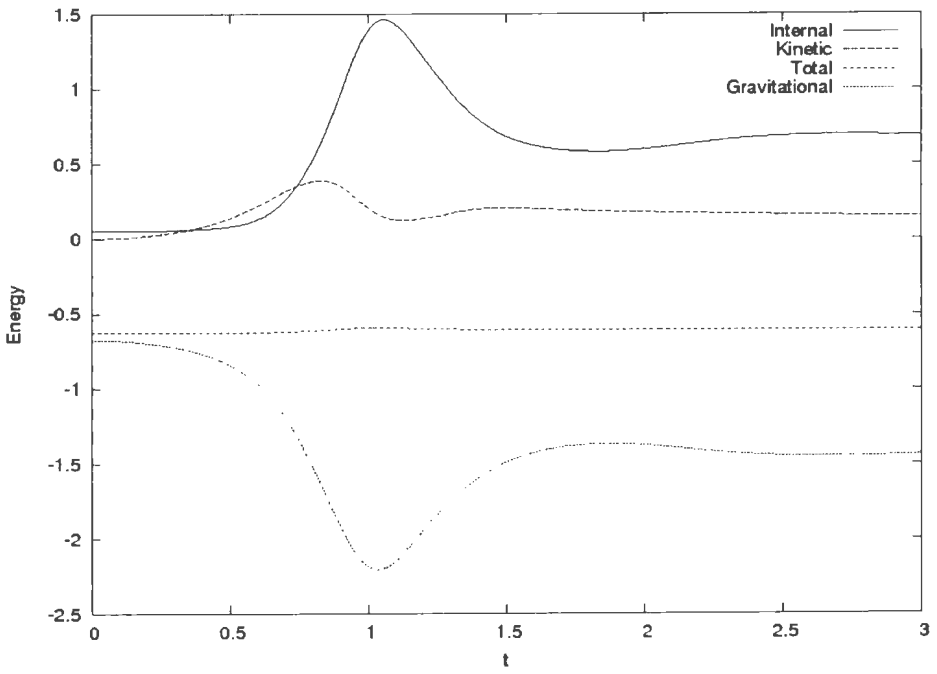

Figure 7.1: System energies of the gas collapse test case. Around $t = 1$, the forces from thermal pressure begin to exceed gravitational forces, and the gas cloud stops contracting.

technique. Starting from a uniform cubic lattice, each particle is displaced radially inwards by

$$r_a^{\text{new}} = r_a^{\text{old}} \left(\frac{r_a^{\text{old}}}{R} \right). \tag{7.2}$$

This creates a density profile proportional to r^{-1}. To achieve the exact profile as in equation 7.1, we modify the displacement function to be

$$r_a^{\text{new}} = r_a^{\text{old}} \left(\frac{(1-a) r_a^{\text{old}} + a}{R} \right). \tag{7.3}$$

where a can be tuned to achieve to profile required.

The results presented are for a run using 1.02208×10^5 particles. Figure 7.1.2 shows

Figure 7.2: The "light" colour scale used for colour images. Black corresponds to the minimum values of the image, with white being the maximum values.

the evolution of internal, kinetic, gravitational, and total energies of the system up to time $t = 3.0$. We can see the internal energy and gravitational energy extrema coincide with each other, as this corresponds to the point of maximum compression and peak temperature. There is slight non-conservation of the total energy around this peak, which arises from the errors involved in using the BH octree to calculate gravitational forces. Total energy is otherwise conserved well.

7.2 Magnetohydrodynamics

Three main test cases were used for the magnetohydrodynamics. The colour images presented in this section use the "light" colour scale in figure 7.2, where black is the minimum value and white is the maximum value.

7.2.1 Fast Rotor

The first test of the MHD code is the two dimensional fast rotor problem introduced by Balsara and Spicer [3]. It consists of a rotating dense disc of fluid situated in an ambient background medium, with a uniform magnetic field threading the system. As the disc rotates, the centrifugal forces cause it spread outwards, but this rotation also wind the magnetic fields inside it which confine it into an oblate shape.

The values used follow that of the first rotor case of Tóth [53]. The disc, with radius $r_0 = 0.1$, is situated in the center of a periodic domain of $-0.5 < x, y < 0.5$. It has a density $\rho_d = 10$ with angular velocity $\omega = 20$, while the background is at rest

with $\rho_b = 1$. The disc and background are set in pressure equilibrium $P = 1$, and the adiabatic index is chosen to be $\gamma = 1.4$. The magnetic field is oriented along the x-axis with $B_x = 5/\sqrt{4\pi}$.

The ambient fluid particles are distributed using a regular square lattice, with the region $r < r_0$ excluded. This area is filled with particles of the same mass also on a square lattice, but with smaller inter-particle seperation to achieve the higher density of the disc. Tóth uses a taper function between the disc and background to reduce the impact of the initial density contrast between the two. However, for our tests the taper function was not used, as the smoothing of the densities present in SPH already compensate for the density contrast to a certain extent.

Presented are the results for a system containing 7.21296×10^5 particles, of which 1.76488×10^5 particles comprise the disc. Figures 7.3 and 7.5 show the thermal and magnetic pressure of the system at $t = 0.1$, and may be compared with the results from Dolag and Stasyszyn [13], while figures 7.4 and 7.6 are at $t = 0.15$ and may be compared with Tóth [53] and Price and Monaghan [43]. A cut of the density following the diagonal $x = y$ through the system at $t = 0.1$ is presented in figure 7.7 and can be compared with Dolag and Stasyszyn [13]. The features of the thermal and magnetic pressure correspond quite well for both times examined, and the density cut is reproduced almost precisely.

7.2.2 Strong Blast

A second test problem presented by Balsara and Spicer is the modeling of a strong hydrodynamic blast wave in a magnetohydrodynamic system. A circular region of a homogeneous medium is increased in pressure a hundred fold and then released. An initially uniform magnetic field is present, which affects the resultant shock wave to

Figure 7.3: Pressure of the fast rotor test case at $t = 0.1$.

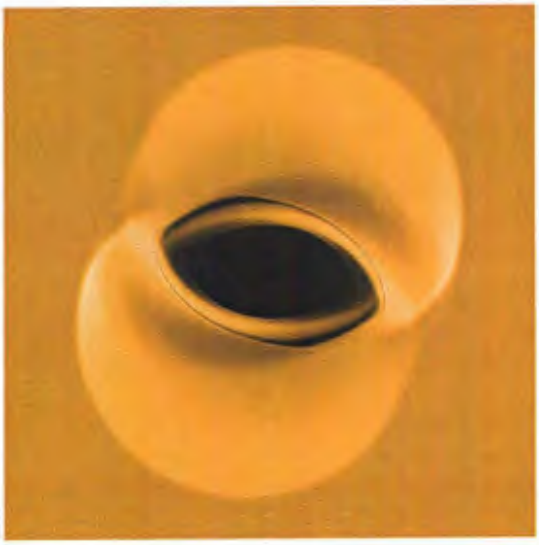

Figure 7.4: Pressure of the fast rotor test case at $t = 0.15$.

Figure 7.5: Magnetic pressure of the fast rotor test case at $t = 0.1$.

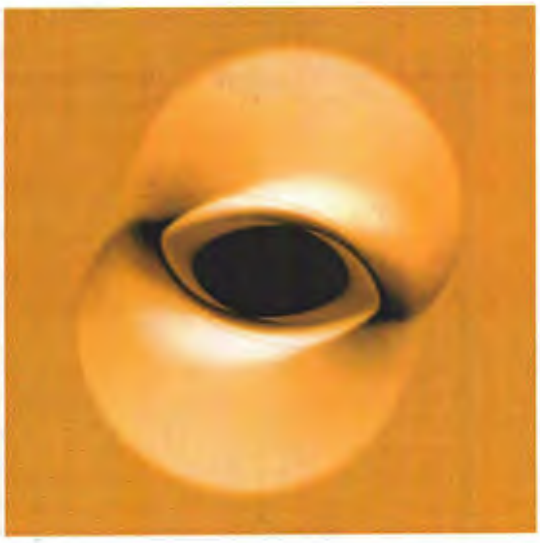

Figure 7.6: Magnetic pressure of the fast rotor test case at $t = 0.15$.

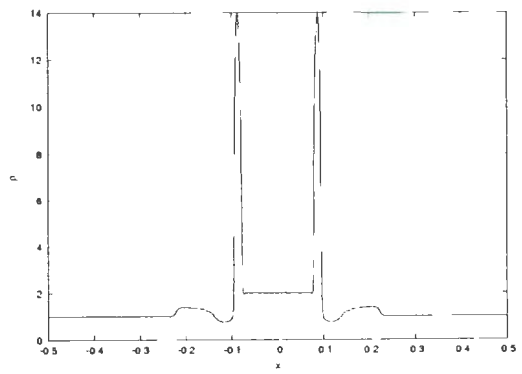

Figure 7.7: Density of the fast rotor test case along a diagonal $x = y$ at $t = 0.1$.

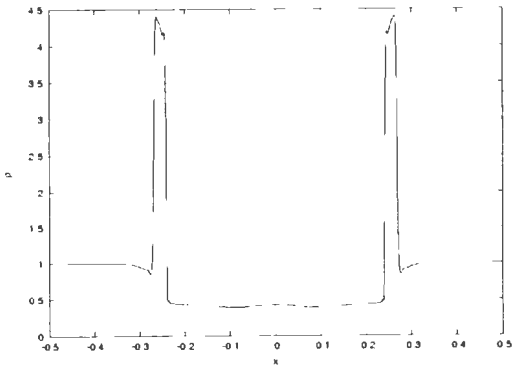

Figure 7.8: Density of the strong blast test case along the horizontal line $y = 0$ at $t = 0.02$.

preferentially propogate along the field lines. The system is initialised according to Londrillo and Del Zanna's specifications [26].

The fluid medium is situated in a periodic domain of $-0.5 < x, y < 0.5$ with density $\rho = 1$. The circular region is chosen at the center with radius $r_o = 0.125$. Its pressure is set to $P_d = 100$, while the outside pressure is $P_b = 1$. The magnetic field is oriented along the x-axis with strength $B_x = 10$.

Reseults are presented for a system of size 7.225×10^5 particles, and are examined at time $t = 0.02$. Figure 7.8 shows a density cut through the middle of the system along the x-axis, which may be compared with Dolag and Stasyszyn [13]. There is a discrepancy in the height of the two peaks, but this can be explained by the stronger viscosity term they use. Figures 7.9 and 7.10 showcase the density and magnetic pressure distributions for the system, which corresponds well to Dolag and Stasyszyn [13] and Londrillo and Del Zanna [26].

Figure 7.9: Density of the strong blast test case.

Figure 7.10: Magnetic pressure of the strong blast test case.

7.2.3 Orszag-Tang Vortex

The final MHD test performed is of the compressible Orszag-Tang vortex, first introduced by Orszag and Tang [36] and is extensively used as a test problem for multi-dimensional MHD codes. The initial flow is given by a velocity vortex superimposed to a magnetic vortex which share a common (singular) X-point, but with different model structure. This unstable configuration produces several classes of shock waves which interact with each other.

The system is situated inside a periodic box of size $0 < x, y < 1$. The velocity field is initialised to $v_x = -\sin(2\pi y)$ and $v_y = \sin(2\pi y)$, and the magnetic field has initial configuration $B_x = B_o v_x$ and $B_y = B_o \sin(4\pi x)$, where $B_o = 1/\sqrt{4\pi}$. The initial gas state is set at $P = \gamma B_o^2$, $\rho = \gamma P$, with $\gamma = 5/3$.

Presented are the results for a system of size 7.225×10^5 particles. Figures 7.11 and 7.12 show the density and magnetic pressure distributions, which compares well with results from Price and Monaghan [43], Rosswog and Price [45], Londrillo and

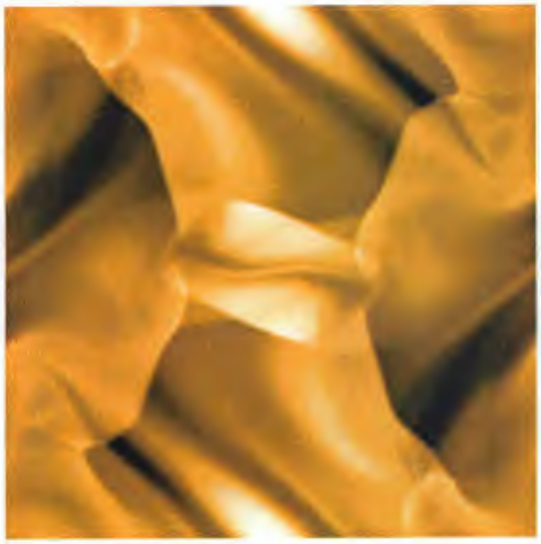

Figure 7.11: Density of the Orszag-Tang vortex at $t = 0.5$.

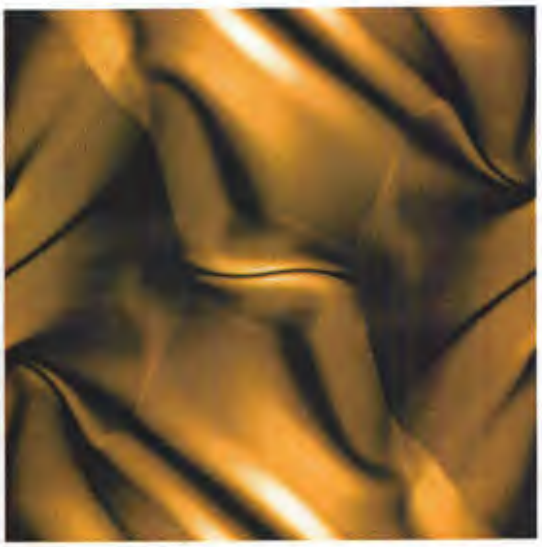

Figure 7.12: Magnetic pressure of the Orszag-Tang vortex at $t = 0.5$.

Del Zanna [26], Dolag and Stasyszyn [13], and Borve *et al.* [9]. Pressure cuts along the horizontal lines $y = 0.3125$ and $y = 0.4277$ are presented in figures 7.13 and 7.14, which exhibit excellent agreement with the results from Dolag and Stasyszyn [13], Borve *et al.* [9], Londrillo and Del Zanna [26], and Rosswog and Price [45].

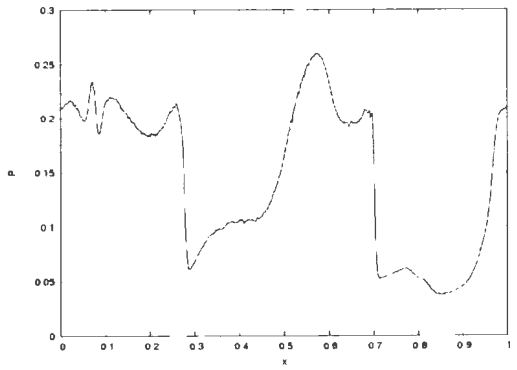

Figure 7.13: Pressure of the Orszag-Tang vortex along the horizontal line $y = 0.3125$ at $t = 0.5$.

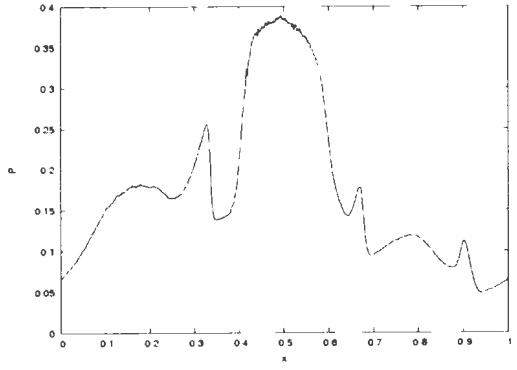

Figure 7.14: Pressure of the Orszag-Tang vortex along the horizontal line $y = 0.4277$ at $t = 0.5$.

"Hello. Farewell. Hello. Farewell."

Tralfamadorian greeting

8

Summary

This thesis has described the theory of magnetohydrodynamics and the numerical scheme for its implementation. A simulation software has been developed for research involving astrophysical plasmas, and also for continued research into the numerical methods of simulating plasmas. The operator design philosophy of this software allows for the ideal environment to develop new numerical methods.

The nearest neighbour search algorithm devised has significant efficiency gains over previously employed methods. For the test cases examined, a several times decrease in search runtime was exhibited. Removal of the $\lg N$ scaling factor allows the efficiency of this algorithm to grow more pronounced as the system size increases. This search algorithm was also modified for the purposes of determining smoothing lengths, which enabled this task to be performed with a consistency of runtime that is not matched by iterative methods.

Finally, visualisation techniques were investigated, which used the SPH particles as vertices in a Delaunay triangulation. Data values from the particles serve as

weightings with which to colour the triangles.

8.1 Future Work

There are several avenues in which this work can be continued. First, the simulation software itself can be improved in a couple of ways. Presently, the software executes serially, and parallelising the code would be of benefit. The scripting used for initialisation could also be extended to allow users to script the flow of the whole program, truly extracting the full power that scripting languages bring.

The second avenue is to improve upon the algorithms of SPH. Development of a code to perform SPH on GPU's (Graphics Processing Units) is highly desireable since their computational capacity far exceeds that of desktop CPU's. In terms of SPMHD, there are a number of issues that are unresolved. The negative stress from magnetic forces has been circumvented, but at the cost of momentum conservation. How can this be fixed? The div $\boldsymbol{B}$ constraint has been upheld through the use of Euler potentials, but introduces restrictions on the types of fields capable of representation. Is there a better way?

Future research is planned to improve the simulation software, and to investigate solutions to the problems regarding SPMHD.

Appendix A

Optimal Values for Viscosity Parameters

The artificial viscosity and resistivity parameters, β, α, and α_B, were examined in greater detail to find optimal values for MHD simulation. The Orszag-Tang vortex problem served as the test bed owing to its complexity. The initial conditions described in section 7.2.3 were used, with the simulations run at 1.6×10^5 particles. Pressure cuts along the x-axis for y=0.3215 were used as the measure of quality.

The sixteen permutations of the values β = [1.0, 1.5, 2.0, 3.0], α = [1.0, 1.5], and α_B = [0.5, 1.0] were tested, with follow up tests for α_B = [0.1, 0.2, 0.3, 0.4]. These values were chosen for their previous use in the literature. Monaghan [28] suggested to use β = 3.0 and α = 1.5, but these values were for his form of the artificial viscosity and for pure hydrodynamic systems. Price and Monaghan [43] redesigned the artificial viscosity for multidimensional MHD systems, where they suggest to use β = [1.0, 2.0], α = 1.0, α_B = 1.0, though they use a switch to control the strength of α and α_B. Dolag and Stasyszyn [13] pair Monaghan's artificial viscosity with Price and Monaghan's artificial resistivity formulation, and find that β = 1.5, α = 2.0, and α_B

= 0.1 works best. Given the range of values cited, it was decided to examine which set of values provided optimal results.

A.1 β Values

The parameter β appears solely in the definition of the signal velocity, but the signal velocity is used for both viscosity and resistivity. When β is varied between the four values [1.0, 1.5, 2.0, 3.0], there appears to be minimal difference in results. Due to this seemingly indifference, it was decided to use a middleground value of $\beta = 2.0$ for our simulations.

A.2 α Values

The values for α, which controls the strength of the artificial viscosity, does have significant impact on the system dynamics. Increasing the value of α from 1.0 to 1.5 results in the lowering of the central peak, which only serves to deviate the result further from the expected. Thus, the lower value of 1.0 was chosen to be used.

A.3 α_β Values

The resistivity parameter α_β was found to give better results for the lower value. As Dolag and Stasyszyn had used values even lower than 0.5, the set of tests was extended to examine the range α_β = [0.1, 0.2, 0.3, 0.4] for β = 2.0, α = 1.0. The pressure cuts through the system show increased quality as α_β is decreased, with optimal results at 0.1. At this level, not only do the peaks and dips of the pressure cut align with expected values, but the small range oscillation between x = 0.0 to 0.1

begins to emerge. Hence, the value of α_B in simulations was chosen to be 0.1.

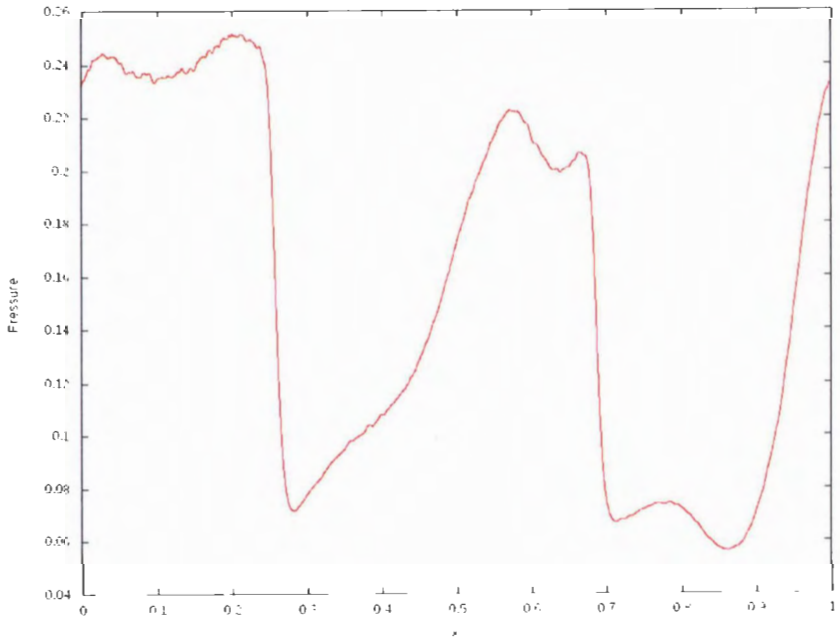

Figure A.1: Orszag-Tang vortex for $\beta = 3.0$, $\alpha = 1.5$, $\alpha_B = 1.0$

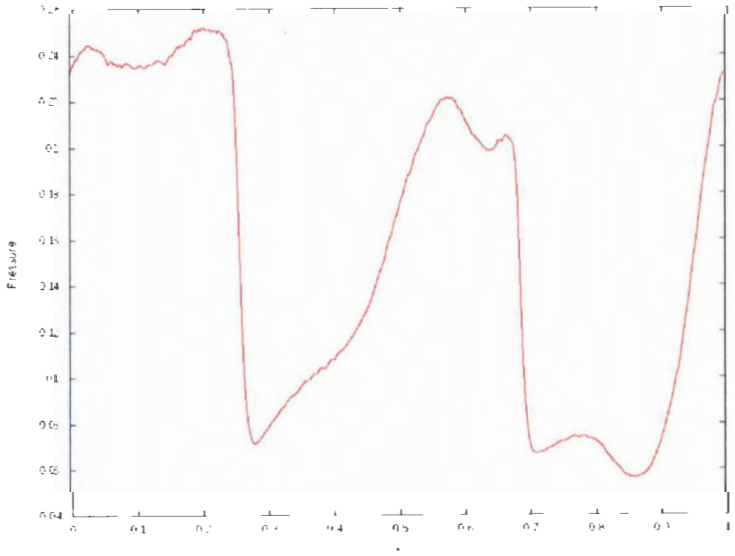

Figure A.2: Orszag-Tang vortex for $\beta = 2.0$, $\alpha = 1.5$, $\alpha_B = 1.0$

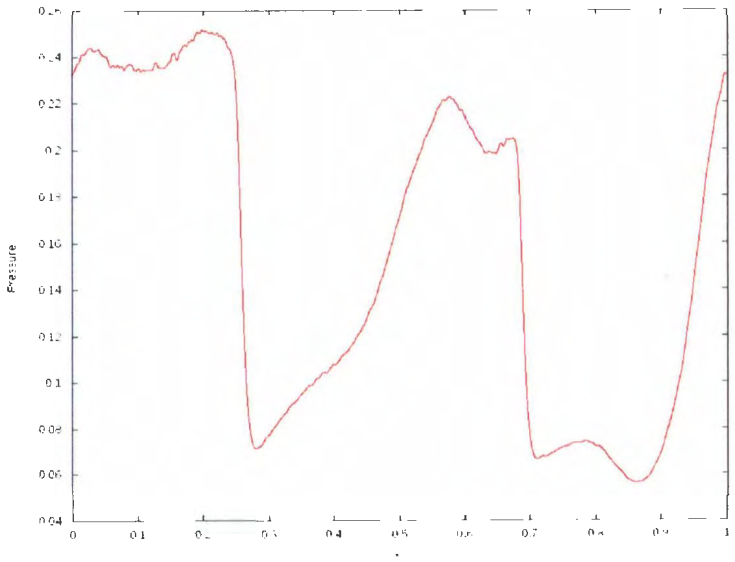

Figure A.3: Orszag-Tang vortex for $\beta = 1.5$, $\alpha = 1.5$, $\alpha_B = 1.0$

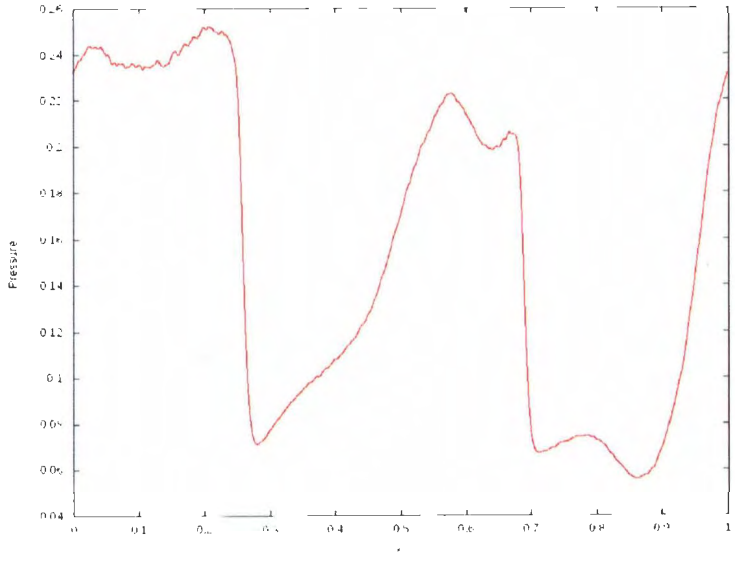

Figure A.4: Orszag-Tang vortex for $\beta = 1.0$, $\alpha = 1.5$, $\alpha_B = 1.0$

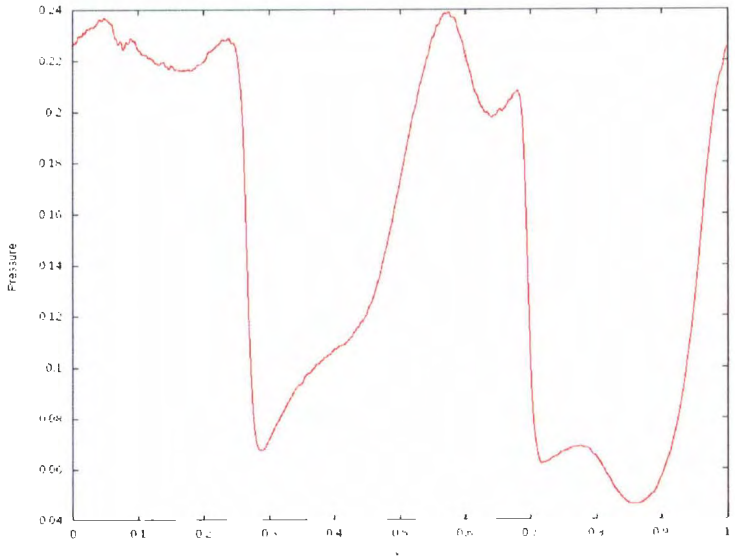

Figure A.5: Orszag-Tang vortex for $\beta = 3.0$, $\alpha = 1.0$, $\alpha_B = 1.0$

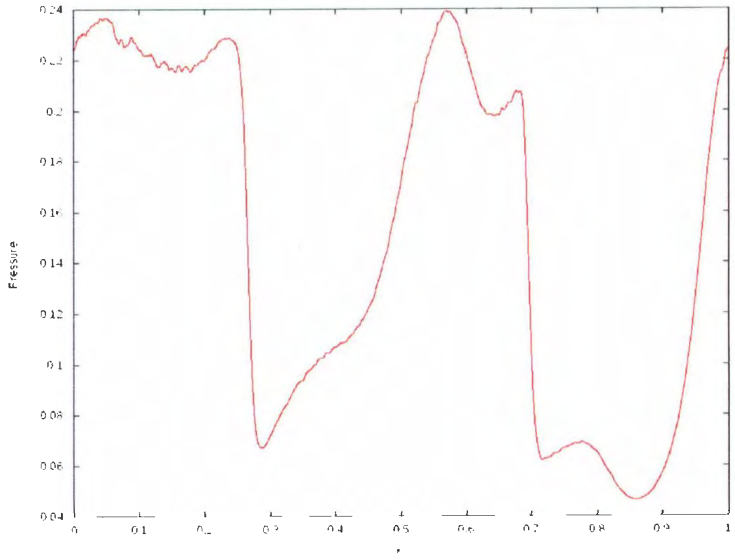

Figure A.6: Orszag-Tang vortex for $\beta = 2.0$, $\alpha = 1.0$, $\alpha_B = 1.0$

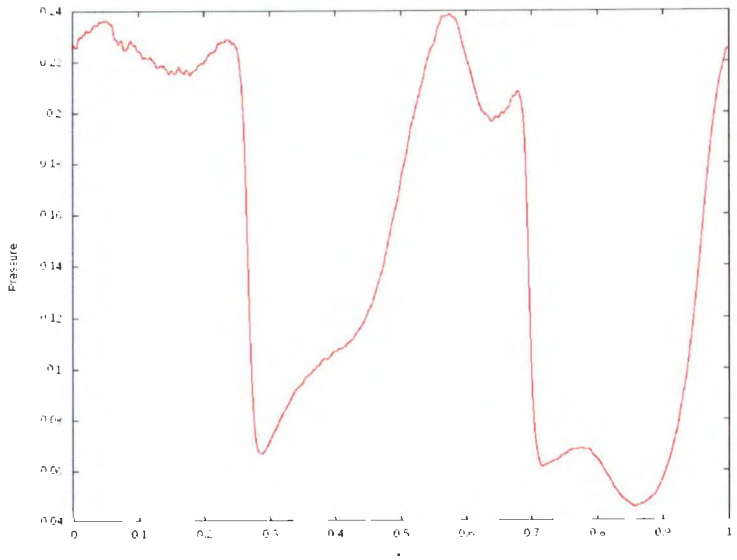

Figure A.7: Orszag-Tang vortex for $\beta = 1.5$, $\alpha = 1.0$, $\alpha_B = 1.0$

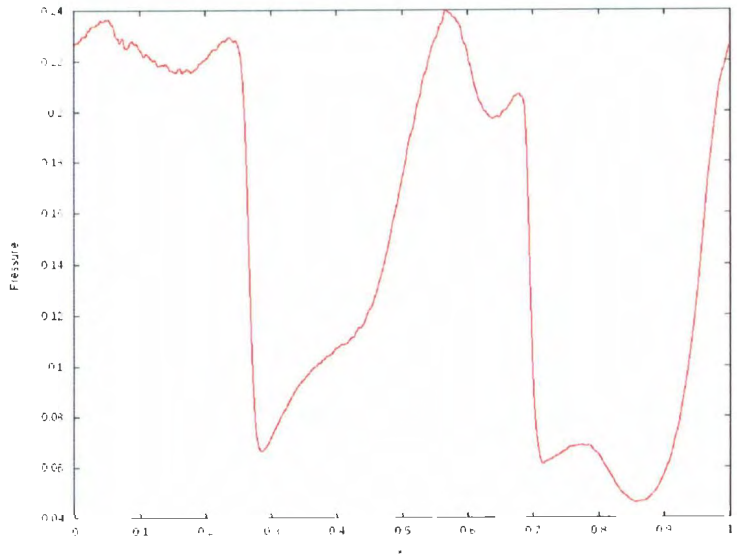

Figure A.8: Orszag-Tang vortex for $\beta = 1.0$, $\alpha = 1.0$, $\alpha_B = 1.0$

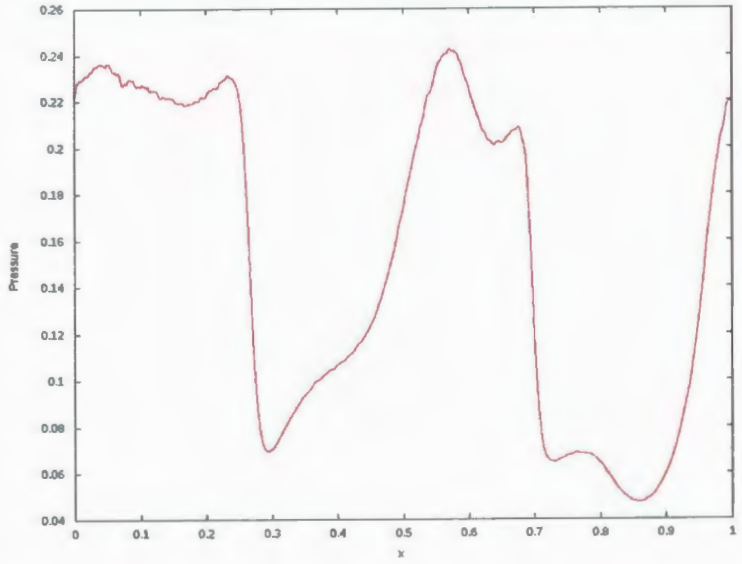

Figure A.9: Orszag-Tang vortex for $\beta = 3.0$, $\alpha = 1.0$, $\alpha_B = 0.5$

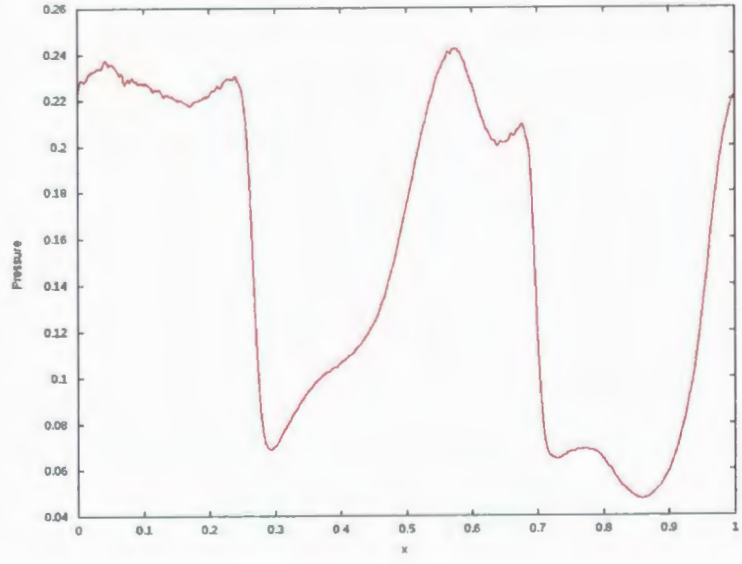

Figure A.10: Orszag-Tang vortex for $\beta = 2.0$, $\alpha = 1.0$, $\alpha_B = 0.5$

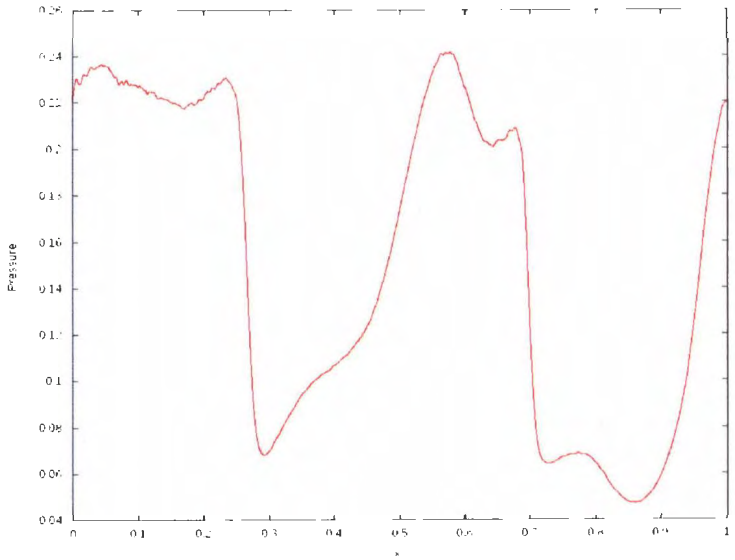

Figure A.11: Orszag-Tang vortex for $\beta = 1.5$, $\alpha = 1.0$, $\alpha_B = 0.5$

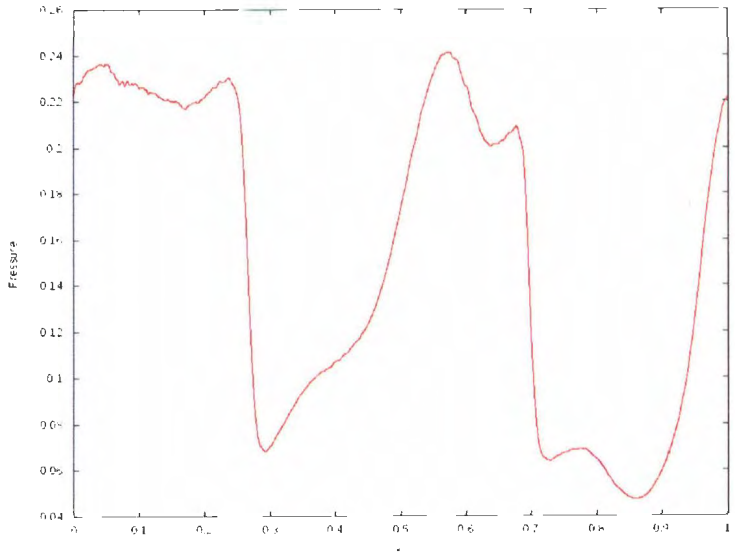

Figure A.12: Orszag-Tang vortex for $\beta = 1.0$, $\alpha = 1.0$, $\alpha_B = 0.5$

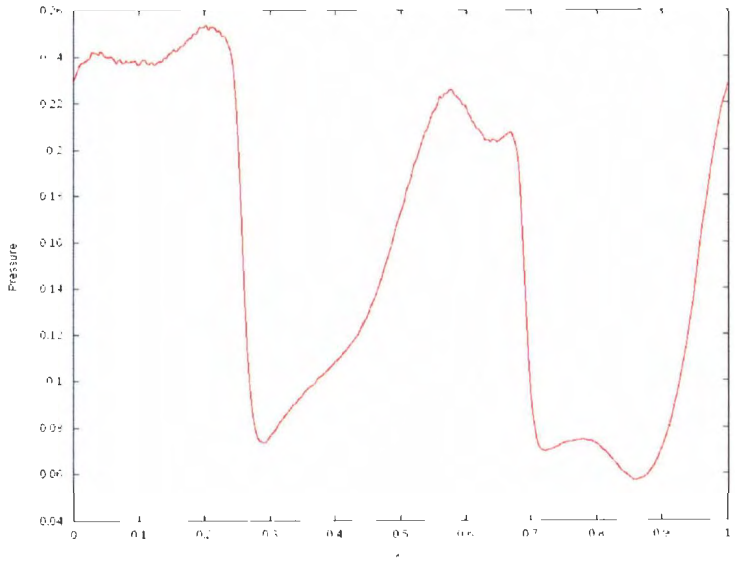

Figure A.13: Orszag-Tang vortex for $\beta = 3.0$, $\alpha = 1.5$, $\alpha_B = 0.5$

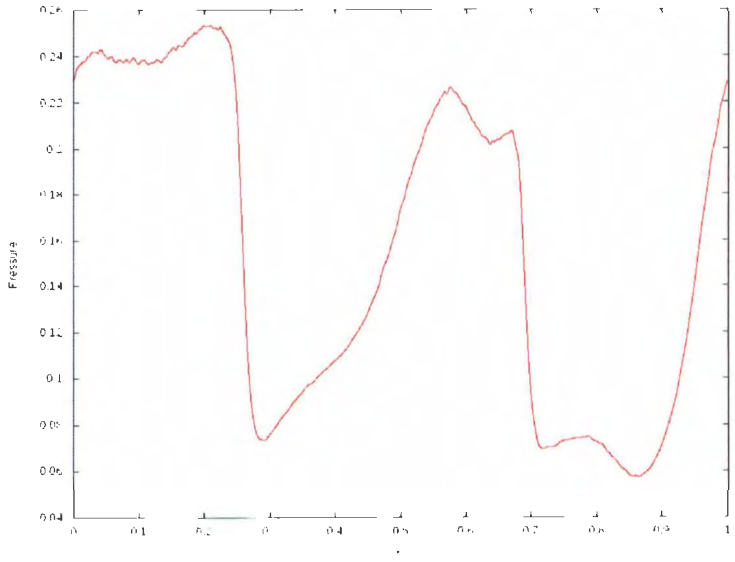

Figure A.14: Orszag-Tang vortex for $\beta = 2.0$, $\alpha = 1.5$, $\alpha_B = 0.5$

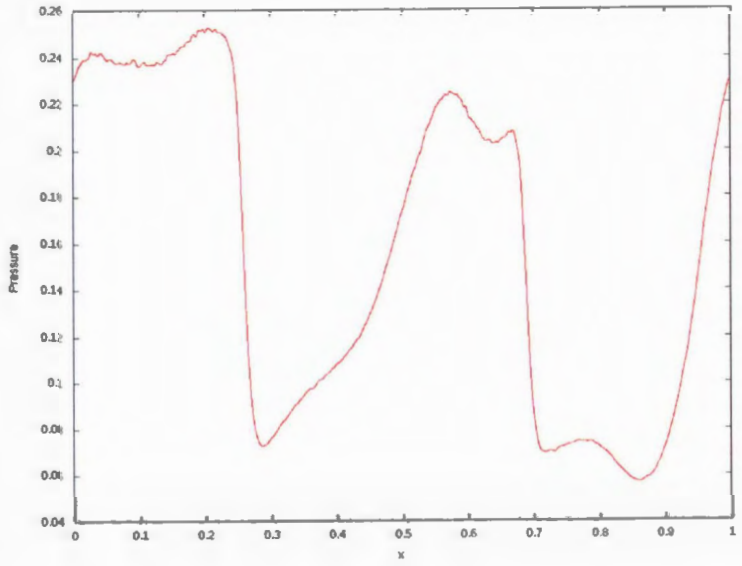

Figure A.15: Orszag-Tang vortex for $\beta = 1.5$, $\alpha = 1.5$, $\alpha_B = 0.5$

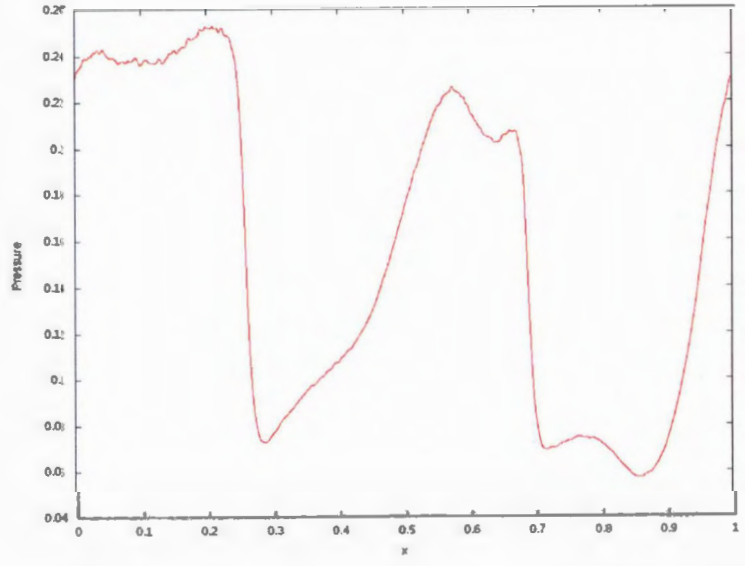

Figure A.16: Orszag-Tang vortex for $\beta = 1.0$, $\alpha = 1.5$, $\alpha_B = 0.5$

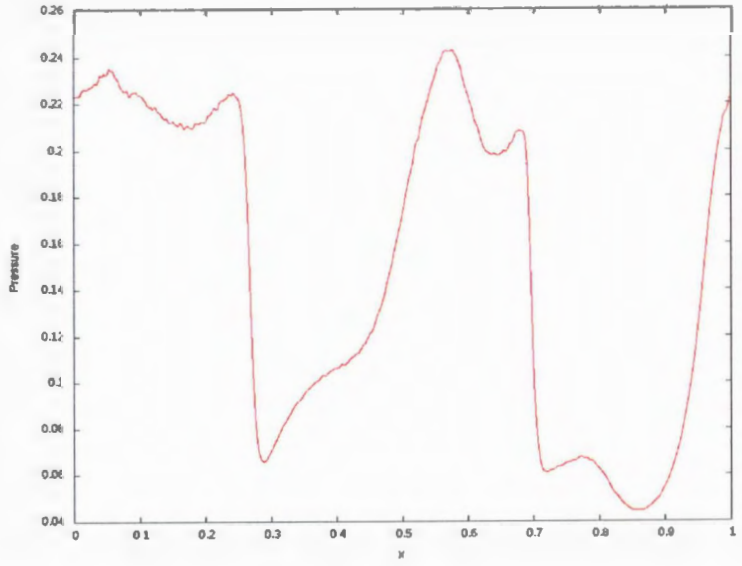

Figure A.17: Orszag-Tang vortex for $\beta = 2.0$, $\alpha = 1.0$, $\alpha_B = 0.4$

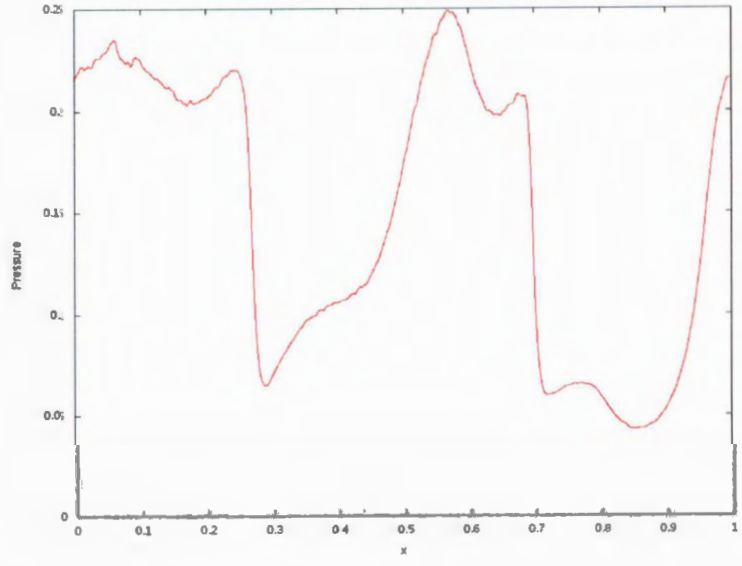

Figure A.18: Orszag-Tang vortex for $\beta = 2.0$, $\alpha = 1.0$, $\alpha_B = 0.3$

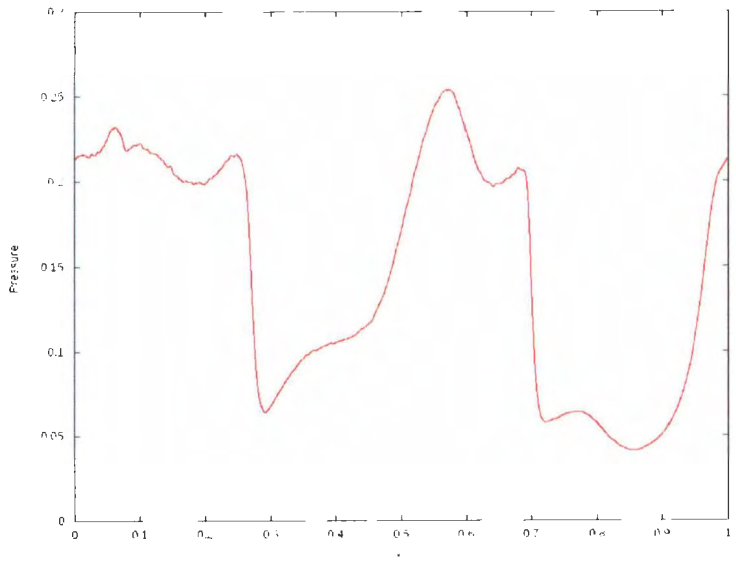

Figure A.19: Orszag-Tang vortex for $\beta = 2.0$, $\alpha = 1.0$, $\alpha_B = 0.2$

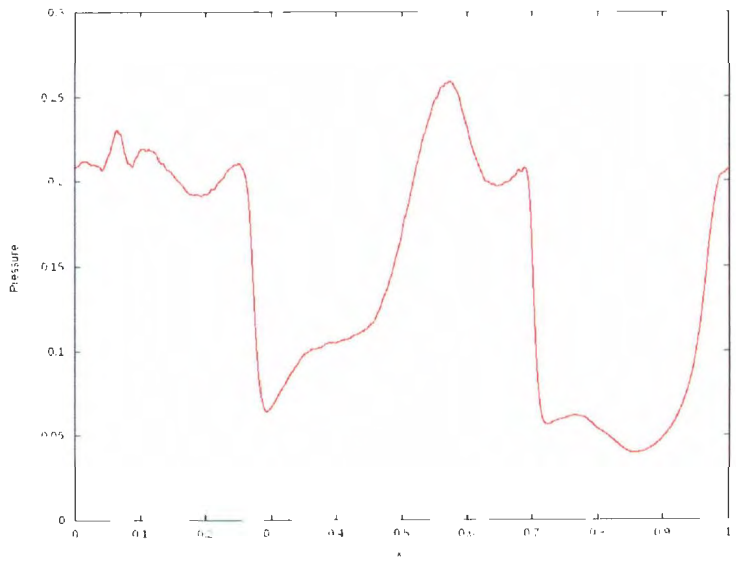

Figure A.20: Orszag-Tang vortex for $\beta = 2.0$, $\alpha = 1.0$, $\alpha_B = 0.1$

Bibliography

[1] J.-M. Alimi, A. Serna, C. Pastor, and G. Bernabeu. Smooth particle hydrodynamics: importance of correction terms in adaptive resolution algorithms. *Journal of Computational Physics*, 192:157–174, November 2003.

[2] D. S. Balsara. von Neumann stability analysis of smooth particle hydrodynamics–suggestions for optimal algorithms. *Journal of Computational Physics*, 121:357–372, 1995.

[3] D. S. Balsara and D. S. Spicer. A Staggered Mesh Algorithm Using High Order Godunov Fluxes to Ensure Solenoidal Magnetic Fields in Magnetohydrodynamic Simulations. *Journal of Computational Physics*, 149:270–292, March 1999.

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

[5] P. M. Bellan. *Fundamentals of Plasma Physics.* Cambridge University Press, 2006.

[6] W. Benz. *The Numerical modelling of nonlinear stellar pulsations : problems and prospects.* Kluwer Academic Publishers, 1990.

[7] J. A. Bittencourt. *Fundamentals of Plasma Physics.* Springer-Verlag New York, Inc., 3rd edition, 2004.

[8] S. Børve, M. Omang, and J. Trulsen. Regularized Smoothed Particle Hydrodynamics: A New Approach to Simulating Magnetohydrodynamic Shocks. ApJ, 561:82 93, November 2001.

[9] S. Børve, M. Omang, and J. Trulsen. Multidimensional MHD Shock Tests of Regularized Smoothed Particle Hydrodynamics. ApJ, 652:1306 1317, December 2006.

[10] T. J. M. Boyd and J. J. Sanderson. *The Physics of Plasmas*. Cambridge University Press, 2003.

[11] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 2nd edition, 2001.

[12] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 3rd edition, 2008.

[13] K. Dolag and F. Stasyszyn. An MHD GADGET for cosmological simulations. MNRAS, 398:1678 1697, October 2009.

[14] A. E. Evrard. Beyond N-body - 3D cosmological gas dynamics. MNRAS, 235:911–934, December 1988.

[15] E. Fehlberg. Low-order classical runge-kutta formules with stepsize control and their application to some heat transfer problems. *NASA Technical Reports*, 315, 1969.

[16] Qt Development Frameworks. Qt. `http://qt.nokia.com`.

[17] R. A. Gingold and J. J. Monaghan. Smoothed particle hydrodynamics - Theory and application to non-spherical stars. MNRAS, 181:375 389, November 1977.

[18] D. A. Gurnett and A. Bhattacharjee. *Introduction to Plasma Physics.* Cambridge University Press, 2005.

[19] E. Hairer, C. Lubich, and G. Wanner. *Geometric Numerical Integration: Structure Preserving Algorithms for Ordinary Differential Equations.* Springer, 2nd edition, 2006.

[20] L. Hernquist. Performance characteristics of tree codes. ApJS, 64:715 734, August 1987.

[21] L. Hernquist. Some cautionary remarks about smoothed particle hydrodynamics. ApJ, 404:717 722, February 1993.

[22] L. Hernquist and N. Katz. TREESPH - A unification of SPH with the hierarchical tree method. ApJS, 70:419 446, June 1989.

[23] R. Ierusalimschy, W. Celes, and de Figueiredo. L. H. Lua. `http://www.lua.org`.

[24] H. Kotarba, H. Lesch, K. Dolag, T. Naab, P. H. Johansson, and F. A. Stasyszyn. Magnetic field structure due to the global velocity field in spiral galaxies. MNRAS, 397:733 747, August 2009.

[25] L. D. Landau and E. M. Lifshitz. *Fluid Mechanics.* Pergamon Press, 3rd edition, 1966.

[26] P. Londrillo and L. Del Zanna. High-Order Upwind Schemes for Multidimensional Magnetohydrodynamics. ApJ, 530:508 524, February 2000.

[27] L. B. Lucy. A numerical approach to the testing of the fission hypothesis. AJ, 82:1013 1024, December 1977.

[28] J. J. Monaghan. SPH and Riemann Solvers. *Journal of Computational Physics*, 136:298 307, September 1997.

[29] J. J. Monaghan. SPH compressible turbulence. MNRAS, 335:843 852, September 2002.

[30] J. J. Monaghan and R. A. Gingold. Shock Simulation by the Particle Method SPH. *Journal of Computational Physics*, 52:374 389, November 1983.

[31] J. J. Monaghan and J. C. Lattanzio. A refined particle method for astrophysical problems. A&A, 149:135 143, August 1985.

[32] J. P. Morris. *Analysis of Smoothed Particle Hydrodynamics with Applications*. PhD thesis, Monash University, 1996.

[33] J. P. Morris and J. J. Monaghan. A Switch to Reduce SPH Viscosity. *Journal of Computational Physics*, 136:41 50, September 1997.

[34] R. P. Nelson and J. C. B. Papaloizou. Three-Dimensional Hydrodynamic Simulations of Collapsing Prolate Clouds. MNRAS, 265:905 +, December 1993.

[35] R. P. Nelson and J. C. B. Papaloizou. Variable Smoothing Lengths and Energy Conservation in Smoothed Particle Hydrodynamics. MNRAS, 270:1-20, September 1994.

[36] S. A. Orszag and C.-M. Tang. Small-scale structure of two-dimensional magnetohydrodynamic turbulence. *Journal of Fluid Mechanics*, 90:129 143, January 1979.

[37] G. J. Phillips and J. J. Monaghan. A numerical method for three-dimensional simulations of collapsing, isothermal, magnetic gas clouds. MNRAS, 216:883 895, October 1985.

[38] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction.* Springer-Verlag, 1985.

[39] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes: The Art of Scientific Computing.* Cambridge University Press, 3rd edition, 2007.

[40] D. J. Price. splash: An Interactive Visualisation Tool for Smoothed Particle Hydrodynamics Simulations. *Publications of the Astronomical Society of Australia*, 24:159 173, October 2007.

[41] D. J. Price and M. R. Bate. The impact of magnetic fields on single and binary star formation. MNRAS, 377:77 90, May 2007.

[42] D. J. Price and J. J. Monaghan. Smoothed Particle Magnetohydrodynamics - I. Algorithm and tests in one dimension. MNRAS, 348:123 138, February 2004.

[43] D. J. Price and J. J. Monaghan. Smoothed Particle Magnetohydrodynamics - III. Multidimensional tests and the $\nabla.B= 0$ constraint. MNRAS, 364:384 406, December 2005.

[44] S. Rosswog, M. B. Davies, F. K. Thielemann, and T. Piran. Merging neutron stars: asymmetric systems. A&A, 360:171 184, August 2000.

[45] S. Rosswog and D. Price. MAGMA: a three-dimensional, Lagrangian magnetohydrodynamics code for merger applications. MNRAS, 379:915 931, August 2007.

[46] J. K. Salmon and M. S. Warren. Skeletons from the treecode closet. *Journal of Computational Physics*, 111:136 155, March 1994.

[47] H. Samet. *The Design and Analysis of Spatial Data Structures.* Addison-Wesley, 1990.

[48] A. Serna, J.-M. Alimi, and J.-P. Chieze. Adaptive Smooth Particle Hydrodynamics and Particle-Particle Coupled Codes: Energy and Entropy Conservation. ApJ, 461:884 +, April 1996.

[49] V. Springel. The cosmological simulation code GADGET-2. MNRAS, 364:1105 1134, December 2005.

[50] V. Springel and L. Hernquist. Cosmological smoothed particle hydrodynamics simulations: the entropy equation. MNRAS, 333:649 664, July 2002.

[51] D. P. Stern. The Motion of Magnetic Field Lines. *Space Science Reviews*, 6:147 173, November 1966.

[52] D. P Stern. Representation of magnetic fields in space. *Reviews of Geophysics and Space Physics*, 14:199 214, May 1976.

[53] G. Tóth. The ∇.B=0 Constraint in Shock-Capturing Magnetohydrodynamics Codes. *Journal of Computational Physics*, 161:605 652, July 2000.

[54] J. W. Wadsley, J. Stadel, and T. Quinn. Gasoline: a flexible, parallel implementation of TreeSPH. *New Astronomy*, 9:137 158, February 2004.

[55] D. R. Williams. Nasa planetary fact sheet. `http://nssdc.gsfc.nasa.gov/planetary/factsheet/`.

[56] T. Williams and C. Kelley. gnuplot. `http://gnuplot.sourceforge.net`.