SPHerical: A PYTHON PACKAGE FOR EXPLORING ALTERNATIVE DARK MATTER MODELS

REUBEN DAVID BLAFF

A THESIS SUBMITTED TO THE FACULTY
OF GRADUATE STUDIES IN PARTIAL
FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN PHYSICS AND ASTRONOMY
YORK UNIVERSITY
TORONTO, ONTARIO

DECEMBER 2021

ABSTRACT

Cold dark matter (CDM) cosmological N-body simulations have proved to be powerful tools for studying the evolution and structure of the universe. These simulations, however, predict certain small-scale structure that disagrees with observation. They are also computationally expensive. Thus, the majority of efforts have focused upon the vanilla CDM model. There are, however, a variety of DM models beyond CDM, such as self-interacting dark matter (SIDM), which has shown promise in resolving some of the aforementioned small-scale structure problems. Our new Python package, `SPHerical`, which we present in this paper, aims to allow for further exploration of SIDM and other alternative DM models. The package is used to simulate the evolution of isolated spherically symmetric DM halos using a Smoothed Particle Hydrodynamics (SPH) framework and gravothermal fluid model. In this paper, we limit our scope to SIDM and compare `SPHerical`'s results with those of similar simulations.

ACKNOWLEDGEMENTS

In putting together this thesis, I consulted numerous people, articles, and texts. Some of the most helpful were: Jan Karsten Trulsen, Edward Brown, Joseph Monaghan, and Refs. [1–3]. I would like to thank them all, as well as Charlotte Bordt who provided data for our comparisons. But above all, the writing of this work and creation of `SPHerical` would not have been possible without the guidance of my thesis advisor, Sean Tulin. His constant support and encouragement were invaluable throughout this process and profoundly appreciated.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

A mounting body of evidence strongly supports the theory that the everyday atomic matter we are familiar with comprises less than 20% of the total matter content of the Universe, while the rest is in the form of so-called dark matter (DM) [4–8]. The existence of DM has both theoretical and observational motivations, from galactic rotation curves to gravitational lensing, from the Cosmic Microwave Background (CMB) to Big Bang nucleosynthesis (BBN), to name but a handful [9–12]. As of this writing, the nature of DM (its properties and characteristics) is still largely unknown. However, the preponderance of evidence and observation indicates that DM is not baryonic (i.e. not made of any Standard Model particles), but rather some new form of matter [13].

So-called cold dark mater (CDM) is the vanilla model used for DM by many in both simulations and theory [14]. In this model, DM only interacts via gravity, i.e. there are no "dark" forces or interactions. Combining CDM with General Relativity (GR) and Big Bang theory forms a cosmological model aptly named ΛCDM (Λ being Einstein's cosmological constant), which accurately describes the structure of the universe on large (cosmological) scales [15, 16]. Using this ΛCDM model, cosmological N-body simulations (like the well-known Millennium Simulation) have been run, simulating the evolution of the Universe [17]. These simulations tell us that on small (galaxy-sized) scales, DM exists as halos, which can be modelled by a cuspy universal density profile, such as the often-used NFW (Navarro-Frenk-White) profile [18–20].

While cuspy profiles like the NFW fit rotation curves for many galaxies rather well, they are not consistent with the observed rotation curves of many small dwarf spiral galaxies and Milky Way satellite galaxies, as well as some galaxy clusters, which are better fit by cored profiles [21–29]. This is known as the *core-cusp problem*. There are other problems with the ΛCDM model predictions on small scales (e.g. the *Too big to fail* and *Missing satellites* problems), which has sparked discussion among the astrophysics community about the idea of alternative theories to CDM which could potentially resolve these issues [30–36].

The argument in favour of exploring these alternative theories rests upon the fact that our familiar Standard Model contains a plethora of particles and forces, which together can form all sorts of different atoms and molecules. Given this, it is very well possible that the nature of DM is far richer and more complex than what the vanilla CDM model assumes. Also, simulations using self-interacting dark matter (SIDM), a general model where collisions and scatterings between DM particles are allowed, have yielded the very cored density distributions favoured by the aforementioned rotation curves [3, 22, 37–41].

As mentioned above, cosmological N-body simulations are a standard technique employed

by astrophysicists to probe and study the structure of DM halos [42–44]. Typically, these simulations populate a "cosmological box" with $\sim 10^9$ DM "particles" and time-evolve their positions in 3D [17]. These particles do not model individual DM particles, but rather a collection of many DM particles of some fixed mass. In a simulation with a billion particles, each might have a mass of $\sim 10^3\ M_\odot$, while the mass of actual DM particles is believed to be significantly smaller (estimates from observations and different particle physics theories range from $1-10^4$ times the mass of the proton) [7, 45–47]. Over 13.8 Gyr (i.e. the age of the Universe), these particles coalesce and collapse under the force of their mutual gravitational attraction, forming large- and small-scale structures.

The above describes a typical DM-only N-body simulation. There are simulations, however, that also include baryonic matter (BM) [45, 48–50]. They are known as hydrodynamical simulations, because the BM is modelled to obey the equations of hydrodynamics. These equations must be solved along with the evolution of the DM-only component. An often-used computational method for such simulations is known as Smoothed Particle Hydrodynamics (SPH) [2, 51]. The technique amounts to discretizing the BM into particles, just like the sort described above for DM. However, whereas the DM particles only experience the force of gravity, and can thus be described simply by their positions and velocities, the BM particles can undergo both dissipations and collisions, and therefore require an extra quantity to describe their evolution: temperature (or equivalently, energy).

Whether hydrodynamical or not, these sorts of CDM simulations are exceedingly computationally expensive and take an immense amount of time to run (millions of CPU-hours, even on supercomputers) [52, 53]. It is for this reason that the vast majority of simulations have used the vanilla CDM model for the DM in their evolutions, as including additional particle physics increases computation time significantly. Thus, not many other DM models have been incorporated into cosmological N-body simulations. Addressing these issues is the motivation behind this thesis, though we limit our discussion to just the SIDM model, and leave additional DM models to future works.

As described above, SIDM assumes DM particles are allowed to collide and scatter with one another. Just as with BM, self-interactions between DM particles can lead to energy exchange. In the context of a DM halo, this means that heat can be transferred between different regions. These regions being in thermal contact with one another can drive the halo toward thermal equilibrium, and in some cases, if the scattering rate between particles is sufficiently high, gravothermal collapse. Since SIDM experiences collision and dissipation, it is governed by the equations of hydrodynamics, making SPH a suitable computational method for simulating the dynamics of an SIDM halo.

Often, the sorts of SPH N-body simulations discussed previously are carried out in 3D

and simulate the evolution of the Universe, but in this thesis, we will be solving for the dynamics of individual DM halos and making a simplifying assumption that these halos obey spherical symmetry (there is evidence that such an assumption is not far-fetched) [54, 55]. The assumption of spherical symmetry amounts to averaging our hydrodynamic SPH equations over solid angle, thus turning the particles of SPH into concentric shells, and our problem into a 1D one [56]. The advantage of the latter is immense: 3D simulations might use a million (on the low end) to a billion (on the high end) particles, whereas our 1D simulations only require $10^2 - 10^3$. We have put together a Python package called `SPHerical` to carry out the sort of simulations described above.

The remainder of this paper is laid out as follows: in Section 2, we introduce the SPH framework and derive the governing equations for an isolated gravothermal fluid with no additional interactions (first in 3D, and then in a spherically symmetric form). In Section 3, we discuss modifications to these equations that arise from the introduction of effects such as viscosity and self-interactions. In Section 4, we describe how all of the above pieces are tied together in our `SPHerical` package. In Section 5, we present results from `SPHerical` that informed our choice of parameters, and also comparisons with those of papers that employed similar but different frameworks. Finally, in Section 6, we sum up our findings, suggest applications for `SPHerical`, and discuss future plans for the package.

2 SMOOTHED PARTICLE HYDRODYNAMICS

As discussed earlier, SPH is an N-body method used to simulate the evolution of a hydrodynamical system. Such systems obey Euler's equations for a compressible fluid or ideal gas. In this section, we will derive these equations for the case of a self-gravitating fluid (e.g. a DM halo) with no additional interactions. The effects of heat conduction (due to self-interactions) and other phenomena such as viscosity will be added to the equations as additional terms and derived in Section 3.

2.1 EULER'S EQUATIONS

Starting only with the assumption of a compressible fluid/ideal gas, Euler's equations arise from conservation laws related to three quantities: mass density $\rho(\mathbf{r}(t), t)$, velocity $\mathbf{v}(\mathbf{r}(t), t)$, and internal energy per unit mass $e(\mathbf{r}(t), t)$, where t is time and $\mathbf{r}(t)$ is position at time t. In the derivations to follow, though all quantities will be functions of $\mathbf{r}(t)$ and t, these variables will be omitted for the sake of clarity.

2.1.1 CONSERVATION OF MASS

Consider a small yet macroscopic fluid parcel of volume V. It's mass can be expressed as

$$M = \int_V \rho dV. \tag{1}$$

The flux of mass into (or out of) this parcel from the surrounding fluid is

$$-\int_S \rho \mathbf{v} \cdot d\mathbf{S}, \tag{2}$$

where S refers to the surface bounding the fluid parcel. Since mass must be conserved, Eq. (2) must be equivalent to

$$\frac{\partial}{\partial t} \int_V \rho dV. \tag{3}$$

Using Gauss's law (i.e. the divergence theorem), we can re-express Eq. (2) as the following volume integral

$$-\int_V \nabla \cdot (\rho \mathbf{v}) dV. \tag{4}$$

Since Eq. (3) and (4) are equal, we can write

$$\int_V \left(\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) \right) dV = 0. \tag{5}$$

As we are considering an arbitrary fluid parcel, this equation must hold true for any volume V, and thus the integrand must vanish and

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0. \tag{6}$$

This is the conservation of mass equation (also known as the continuity equation).

2.1.2 CONSERVATION OF MOMENTUM

Neglecting viscosity (which we will consider in Section 3), the net force on our fluid parcel from 2.1.1 comes from two sources: the pressure $P(\mathbf{r}(t), t)$ over its surface and its gravitational potential $\Phi(\mathbf{r}(t), t)$, the latter of which obeys Poisson's equation

$$\nabla^2 \Phi = 4\pi G \rho. \tag{7}$$

Thus, we can write the fluid analog of Newton's second law of motion $\mathbf{F} = m\mathbf{a}$ as follows: the left-hand side of the equation can be expressed as

$$\int_V \rho \nabla \Phi dV - \int_S P d\mathbf{S}. \tag{8}$$

Again using the divergence theorem, we can re-write the surface integral in Eq. (8) as

$$-\int_V \nabla P dV. \tag{9}$$

The right-hand side of our fluid analog of Newton's second law takes the form

$$\int_V \left(\rho \frac{d^2 \mathbf{r}}{dt^2} \right) dV. \tag{10}$$

Thus, we arrive at the equation

$$\int_V \left(\rho \frac{d^2 \mathbf{r}}{dt^2} + \rho \nabla \Phi + \nabla P \right) dV = 0. \tag{11}$$

Once again, this equation must hold for any fluid parcel. Thus, the integrand must vanish and therefore

$$\rho \frac{d^2 \mathbf{r}}{dt^2} + \rho \nabla \Phi + \nabla P = 0. \tag{12}$$

Dividing through by ρ and rearranging, we obtain the more familiar form of the equation

$$\frac{d^2\mathbf{r}}{dt^2} = -\nabla\Phi - \frac{1}{\rho}\nabla P. \tag{13}$$

Recall that $\mathbf{v} = \mathbf{v}(\mathbf{r}(t), t)$. $\mathbf{v}$ refers to the velocity field of the fluid we're considering, **not** the velocity of the particular parcel we've chosen to focus on. That is to say, the parcel can still accelerate even if $\partial\mathbf{v}/\partial t = 0$ by simply moving to a new position. At time t, our parcel at position $\mathbf{r}$ has velocity

$$\left.\frac{\partial\mathbf{r}}{\partial t}\right|_t = \mathbf{v}(\mathbf{r}|_t, t). \tag{14}$$

At an infinitesimally later time $t + h$, the particle has moved to a new location

$$\mathbf{r}(t + h) \approx \mathbf{r}(t) + h\mathbf{v}(\mathbf{r}|_t, t), \tag{15}$$

and its velocity has become

$$\left.\frac{\partial\mathbf{r}}{\partial t}\right|_{t+h} = \mathbf{v}(\mathbf{r}|_{t+h}, t + h) \approx \mathbf{v}(\mathbf{r}|_t, t) + h\left[\mathbf{v}(\mathbf{r}|_t, t) \cdot \nabla\mathbf{v}(\mathbf{r}|_t, t) + \frac{\partial\mathbf{v}(\mathbf{r}|_t, t)}{\partial t}\right], \tag{16}$$

where the term in the square brackets is the total (or Lagrangian/material) derivative of $\mathbf{v}(\mathbf{r}|_t, t)$. Subtracting the right-hand side of Eq. (14) from the right-hand side of Eq. (16) and dividing by h gives the acceleration of our fluid parcel. In mathematical terms, we write

$$\frac{d^2\mathbf{r}}{dt^2} = \lim_{h\to 0} \frac{\frac{\partial\mathbf{r}}{\partial t}|_{t+h} - \frac{\partial\mathbf{r}}{\partial t}|_t}{h} = (\mathbf{v} \cdot \nabla)\mathbf{v} + \frac{\partial\mathbf{v}}{\partial t}. \tag{17}$$

where we have omitted all r and t-dependence for clarity's sake. Thus, we can re-write Eq. (13) as

$$\frac{\partial\mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla)\mathbf{v} = -\nabla\Phi - \frac{1}{\rho}\nabla P. \tag{18}$$

This is the conservation of momentum equation (though it's easier to see that if you multiply through Eq. (18) by ρ and then use Eq. (6)).

2.1.3 CONSERVATION OF ENERGY

The fluid parcel we've been considering thus far possesses two forms of energy: internal and kinetic. We can write its total energy as

$$\int_V \left(\frac{1}{2}\rho v^2 + \rho e\right) dV, \tag{19}$$

where e is the internal energy per unit mass of the fluid parcel and $v^2 = \mathbf{v} \cdot \mathbf{v}$. Therefore, one source of energy flux into (or out of) this volume can be expressed as

$$-\int_S \left(\frac{1}{2}\rho v^2 + \rho e\right) \mathbf{v} \cdot d\mathbf{S}. \tag{20}$$

Another source can come in the form of work done by pressure acting on fluid flowing into our parcel. Mathematically, we write this as

$$-\int_S P\mathbf{v} \cdot d\mathbf{S}. \tag{21}$$

Energy can also enter or leave the fluid parcel via work done by gravity. This flux can be expressed as

$$-\int_V \rho\mathbf{v} \cdot (\nabla\Phi) dV. \tag{22}$$

Thus, we can make the following statement about the time-derivative of Eq. (19)

$$\frac{\partial}{\partial t}\int_V \left(\frac{1}{2}\rho v^2 + \rho e\right) dV = -\int_S \left(\frac{1}{2}\rho v^2 + \rho e + P\right) \mathbf{v} \cdot d\mathbf{S} - \int_V \rho\mathbf{v} \cdot (\nabla\Phi) dV. \tag{23}$$

Once more using the divergence theorem to turn our surface integrals into volume ones, and rearranging Eq. (23), we obtain the following equation

$$\int_V \left\{\frac{\partial}{\partial t}\left(\frac{1}{2}\rho v^2 + \rho e\right) + \nabla \cdot \left[\rho\mathbf{v}\left(\frac{1}{2}v^2 + e + \frac{P}{\rho}\right)\right] + \rho\mathbf{v} \cdot (\nabla\Phi)\right\} dV = 0. \tag{24}$$

Yet again, this equation must be true for any fluid parcel, as our particular one is arbitrary. Thus, the integrand must vanish and therefore

$$\frac{\partial}{\partial t}\left(\frac{1}{2}\rho v^2 + \rho e\right) + \nabla \cdot \left[\rho\mathbf{v}\left(\frac{1}{2}v^2 + e + \frac{P}{\rho}\right)\right] + \rho\mathbf{v} \cdot (\nabla\Phi) = 0. \tag{25}$$

We can expand the time derivative in the above equation as follows

$$\frac{\partial}{\partial t}\left(\frac{1}{2}\rho v^2 + \rho e\right) = \left(\frac{1}{2}v^2 + e\right)\frac{\partial \rho}{\partial t} + \rho\frac{\partial}{\partial t}\left(\frac{1}{2}(\mathbf{v} \cdot \mathbf{v}) + e\right). \tag{26}$$

Using the expression for $\partial\rho/\partial t$ given by Eq. (6), and further expanding, we obtain

$$\frac{\partial}{\partial t}\left(\frac{1}{2}\rho v^2 + \rho e\right) = -\left(\frac{1}{2}v^2 + e\right)\nabla \cdot (\rho\mathbf{v}) + \rho\mathbf{v}\frac{\partial \mathbf{v}}{\partial t} + \rho\frac{\partial e}{\partial t}. \tag{27}$$

Substituting in the expression for $\partial \mathbf{v}/\partial t$ from Eq. (18), and noting $\mathbf{v}(\nabla \cdot \mathbf{v})\mathbf{v} = \mathbf{v} \cdot \nabla \left[u^2/2\right]$, we can re-write Eq. (25) as

$$\rho \frac{\partial e}{\partial t} + \rho \mathbf{v} \cdot \nabla e + P \nabla \cdot \mathbf{v} = 0. \tag{28}$$

Dividing through the above equation by ρ and rearranging, we obtain

$$\frac{\partial e}{\partial t} + \mathbf{v} \cdot \nabla e = -\frac{P}{\rho} \nabla \cdot \mathbf{v}. \tag{29}$$

This is the conservation of energy equation.

2.1.4 EQUATION OF STATE

The above three Euler's equations contain four variables ($\rho, \mathbf{v}, P$, and e), and therefore, we need an additional equation to solve the system. Fortunately, for an ideal gas, we have the relationship

$$P = (\gamma - 1) e \rho, \tag{30}$$

where γ is the heat capacity ratio and $\gamma = \frac{5}{3}$ for monatomic gases (which we will assume our fluid/gas to be).

2.1.5 SOLVING EULER'S EQUATIONS

When it comes to solving Eqs. (6), (18), (29), and (30), there are two predominant numerical approaches: the Eulerian and Lagrangian frameworks.

In the Eulerian approach, a geometric grid of positions $\mathbf{r}$ (it can be fixed or adpative) is constructed and the aforementioned four variables of Euler's equations are defined over the grid cells. Euler's equations are then discretized and solved on this mesh. Solving these equations amounts to determining how our fluid flows into/out of each grid cell over time.

Conversely, in the Lagrangian approach, there is no mesh involved. Instead, the fluid itself is discretized into parcels (or "particles") and solving Euler's equations amounts to tracking the co-moving parcel positions $\mathbf{r}(t)$ and other quantities as a function of time. SPH, the approach we will be considering in this paper, is a Lagrangian method.

While neither method is necessarily "better" than the other, each has its advantages/disadvantages. For example, some Eulerian methods like Adaptive Mesh Refinement (AMR) have higher resolution than SPH when using an equal number of particles/grid cells. On the other hand, SPH handles advection seamlessly, whereas AMR struggles and often yields unphysical results. For a more comprehensive comparison, see the introduction of Ref. [1].

2.2 SPH FUNDAMENTALS

As discussed in 2.1.5, SPH involves dividing up a continuous fluid into a collection of discrete particles. Each particle is labelled by an index i and characterized by four main quantities: its position $\mathbf{r}_i$, its velocity $\mathbf{v}_i = d\mathbf{r}_i/dt$, its specific internal energy e_i, and its mass m_i. It is important to clarify that, in the case of applying SPH to a DM halo, each particle i represents a chunk of the halo and a large number of actual DM particles. It is also worth noting that the quantity e_i reflects the (internal) kinetic motion of these DM particles and $\mathbf{v}_i$ characterizes their bulk motion.

To compute how SPH particle positions evolve with time, we need to solve Newton's second law of motion for each particle

$$\mathbf{F}_i = m_i \frac{d^2\mathbf{r}_i}{dt^2}. \tag{31}$$

We also need to determine the hydrodynamical quantities associated with each particle. To calculate the density at the position of particle i (written as $\rho_i = \rho(\mathbf{r}_i)$), one might consider computing the average density within a sample volume V_i around particle i by summing over neighbouring particles as follows

$$\rho_i = \frac{1}{V_i} \sum_{j \text{ in } V_i} m_j. \tag{32}$$

There are issues with Eq. (32), however. One problem is that this formula is sensitive to the edges of the sample volume. Whether a neighbouring particle j is just inside or outside V_i can have a significant effect on the value of ρ_i, since all particles within the sample volume are "weighted" equally. It makes more sense to have those neighbouring particles j closest to particle i weighed more heavily than more distant neighbours. Another concern with Eq. (32) is that it is not so clear how to take the spatial derivative $\partial\rho_i/\partial\mathbf{r}_i$, which is needed to compute other particle quantities of interest (as we will see below).

2.2.1 SMOOTHING KERNEL

One solution to the above problems is to employ a so-called smoothing kernel, written in most SPH texts as $W(\mathbf{r}, h)$, where h is the kernel's characteristic radius and is referred to as the smoothing length. Here, however, we will denote the kernel by $W_{3\mathrm{D}}(\mathbf{r}, h)$ to differentiate it from the 1D kernel we will define later when we consider the spherically symmetric version of SPH. $W_{3\mathrm{D}}(\mathbf{r}, h)$ is nothing more than a weighting function and characterizes how an SPH particle is localized in space. Effective kernels possess the following properties:

- Maximum at $\mathbf{r} = 0$.

- Only depend on $|\mathbf{r}|$.
- Positive (or zero) for all combinations of $\mathbf{r}$ and h.
- Continuous up to second derivatives everywhere.
- Integral over all space obeys

$$\int d^3\mathbf{r}'\, W_{3\mathrm{D}}(\mathbf{r}', h) = 1. \tag{33}$$

- Approximates a Dirac delta function in the continuum limit $h \to 0$, i.e.

$$\lim_{h\to 0} W_{3\mathrm{D}}(\mathbf{r}, h) \to \delta^3(\mathbf{r}). \tag{34}$$

An example of a suitable kernel is a normalized Gaussian function of the following form

$$W_{3\mathrm{D}}(\mathbf{r}, h) = \frac{1}{(2\pi h^2)^{3/2}} \exp\left(-\frac{|\mathbf{r}|^2}{2h^2}\right). \tag{35}$$

Though this Gaussian obeys the above requirements for a good kernel, it also possess so-called "infinite support," meaning the function never goes to zero. Thus, employing it in SPH frameworks is computationally expensive, since even far-separated particles will "couple" with one another, requiring calculations that will scarcely effect our quantities of interest.

An often-used alternative to this kernel is a so-called "cubic spline" function of the form

$$W_{3\mathrm{D}}(\mathbf{r}, h) = \frac{1}{\pi h^3} \begin{cases} 1 - \frac{3}{2}q^2 + \frac{3}{4}q^3 & 0 \le q < 1 \\ \frac{1}{4}(2-q)^3 & 1 \le q < 2 \\ 0 & q \ge 2 \end{cases}, \tag{36}$$

where $q = |\mathbf{r}|/h$. Not only does this function possess finite (or compact) support, but it is also continuous up to its second derivatives.

In the continuum limit, an arbitrary function $f(\mathbf{r})$ can be expressed as

$$f(\mathbf{r}) = \int d^3\mathbf{r}'\, f(\mathbf{r}')\, \delta^3(\mathbf{r} - \mathbf{r}') \approx \int d^3\mathbf{r}'\, f(\mathbf{r}')\, W_{3\mathrm{D}}(\mathbf{r} - \mathbf{r}', h). \tag{37}$$

However, when we discretize our fluid

$$\int d^3\mathbf{r}' \;\longrightarrow\; \sum_j \frac{m_j}{\rho_j}. \tag{38}$$

Thus, we can write a discretized version of Eq. (37)

$$f(\mathbf{r}) = \sum_j \frac{m_j}{\rho_j} f(\mathbf{r}_j) W_{3\mathrm{D}}(\mathbf{r} - \mathbf{r}_j, h). \tag{39}$$

The power of Eq. (39) is that it allows one to approximate a continuous function by its value at a set of discrete points, i.e. where the SPH particles are located. Thus, we can replace Eq. (32) with

$$\rho_i = \sum_j m_j W_{3\mathrm{D}}(\mathbf{r}_{ij}, h), \tag{40}$$

where $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$. Eq. (39) also makes taking derivatives straightforward. We simply need to differentiate the kernel itself, e.g.

$$\frac{\partial f_i}{\partial \mathbf{r}_i} = \sum_j \frac{m_j}{\rho_j} f_j \frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij}, h)}{\partial \mathbf{r}_i}. \tag{41}$$

Thus, so long as our choice of kernel possesses continuous derivatives, the derivative of our SPH quantities will be smooth and simple to calculate.

2.2.2 EQUATIONS OF MOTION

Now that we can compute ρ and its derivatives for our SPH particles, we are ready to return to Eq. (31). We need an expression for the left-hand side of this formula, i.e. $\mathbf{F}_i$. One approach to deriving this expression is to start from the Euler-Lagrange equations

$$\frac{\partial L}{\partial \mathbf{r}_i} = \frac{d}{dt}\left(\frac{\partial L}{\partial \mathbf{v}_i}\right), \tag{42}$$

where L is the Lagrangian. Recall that $L = T - U$, where T is the kinetic energy and U is the potential energy of a system. In the case of a continuous self-gravitating fluid/gas, the Lagrangian takes the form

$$L = \int d^3\mathbf{r}' \, \rho \left[\frac{1}{2}|\mathbf{v}|^2 - \Phi - e\right]. \tag{43}$$

To obtain the discretized version of Eq. (43), we use Eq. (38), and thus, our new Lagrangian takes the form

$$L = \sum_i m_i \left[\frac{1}{2}|\mathbf{v}_i|^2 - \Phi_i - e_i\right]. \tag{44}$$

If we take our particle masses to be constant ($dm_i/dt = 0$), then plugging this Lagrangian into Eq. (42) yields

$$m_i \mathbf{g}_i - \sum_j m_j \frac{\partial e_j}{\partial \mathbf{r}_i} = m_i \frac{d\mathbf{v}_i}{dt}, \tag{45}$$

where $\mathbf{g}_i = -\partial \Phi_i / \partial \mathbf{r}_i$. The left-hand side of Eq. (45) is precisely $\mathbf{F}_i$. Let us take a closer look at the two terms that make it up. First, we consider $m_i \mathbf{g}_i$. The gravitational potential of particle i is given by

$$\Phi_i = -\sum_{j \neq i} \frac{G m_j}{|\mathbf{r}_{ij}|}. \tag{46}$$

Taking $\partial / \partial \mathbf{r}_i$ of Eq. (46) and multiplying by $-m_i$, we find

$$m_i \mathbf{g}_i = -\sum_{j \neq i} \frac{G m_i m_j}{|\mathbf{r}_{ij}|^3} \mathbf{r}_{ij}. \tag{47}$$

For the term $\sum_j m_j (\partial e_j / \partial \mathbf{r}_i)$, we start by taking the derivative of Eq. (30)

$$\frac{\partial e_j}{\partial \mathbf{r}_i} = \frac{1}{\gamma - 1} \left(\frac{1}{\rho_j} \frac{\partial P_j}{\partial \mathbf{r}_i} - \frac{P_j}{\rho_j^2} \frac{\partial \rho_j}{\partial \mathbf{r}_i} \right). \tag{48}$$

Since entropy is conserved, the relation $P_j \propto \rho_j^\gamma$ holds. Thus, Eq. (48) simplifies to

$$\frac{\partial e_j}{\partial \mathbf{r}_i} = \frac{P_j}{\rho_j^2} \frac{\partial \rho_j}{\partial \mathbf{r}_i}. \tag{49}$$

To get an expression for $\partial \rho_j / \partial \mathbf{r}_i$, we simply take the derivative of Eq. (40)

$$\begin{aligned} \frac{\partial \rho_j}{\partial \mathbf{r}_i} &= \sum_k m_k \frac{\partial W_{\text{3D}}(\mathbf{r}_{jk}, h)}{\partial \mathbf{r}_i} \\ &= \sum_k m_k \left(\frac{\partial W_{\text{3D}}(\mathbf{r}_{ik}, h)}{\partial \mathbf{r}_i} \delta_{ij} - \frac{\partial W_{\text{3D}}(\mathbf{r}_{ji}, h)}{\partial \mathbf{r}_i} \delta_{ik} \right). \end{aligned} \tag{50}$$

Using the fact that $W_{\text{3D}}(\mathbf{r}_{ij}, h) = W_{\text{3D}}(\mathbf{r}_{ji}, h)$, and relabelling some indices, we find

$$\sum_j m_j \frac{\partial e_j}{\partial \mathbf{r}_i} = m_i \sum_j m_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2} \right) \frac{\partial W_{\text{3D}}(\mathbf{r}_{ij}, h)}{\partial \mathbf{r}_i}. \tag{51}$$

Thus, Eq. (45) becomes

$$\frac{d\mathbf{v}_i}{dt} = -\sum_j m_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2} \right) \frac{\partial W_{\text{3D}}(\mathbf{r}_{ij}, h)}{\partial \mathbf{r}_i} - \sum_{j \neq i} \frac{G m_j}{|\mathbf{r}_{ij}|^3} \mathbf{r}_{ij}. \tag{52}$$

This is nothing more than the discretized SPH version of Eq. (18), i.e. the conservation of momentum equation. On the left-hand side, we have the total time-derivative of $\mathbf{v}_i$, and on the right-hand side, the second term is simply $-\nabla_i \Phi_i$, and the first term is the pressure gradient. To see this more clearly, we can use the identity

$$-\frac{1}{\rho}\nabla P = -\frac{P}{\rho^2}\nabla\rho - \nabla\left(\frac{P}{\rho}\right). \tag{53}$$

Computing the right-hand side of this formula using Eq. (41), and evaluating at $\mathbf{r} = \mathbf{r}_i$, we obtain

$$-\frac{1}{\rho}\nabla P\bigg|_{\mathbf{r}=\mathbf{r}_i} = -\frac{P_i}{\rho_i^2}\sum_j m_j \frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij}, h)}{\partial \mathbf{r}_i} - \sum_j m_j \frac{P_j}{\rho_j^2}\frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij}, h)}{\partial \mathbf{r}_i}, \tag{54}$$

which is equivalent to the first term in Eq. (52).

2.2.3 ENERGY EQUATION

Eq. (52) and $\mathbf{v}_i = d\mathbf{r}_i/dt$ make up the equations of motion for our SPH particles. However, we still need to know how e and P evolve with time. To determine this, we start by taking the time-derivative of our equation of state

$$\frac{de_i}{dt} = \frac{1}{\gamma - 1}\left(\frac{1}{\rho_j}\frac{dP_i}{dt} - \frac{P_i}{\rho_i^2}\frac{d\rho_i}{dt}\right) = \frac{P_i}{\rho_i^2}\frac{d\rho_i}{dt}, \tag{55}$$

again using the relationship between P and ρ that holds since entropy is conserved. Plugging in the expression for ρ_i from Eq. (40), we obtain

$$\begin{aligned}\frac{de_i}{dt} &= \frac{P_i}{\rho_i^2}\sum_j m_j \left(\frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij}, h)}{\partial \mathbf{r}_i}\frac{\partial \mathbf{r}_i}{\partial t} + \frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij}, h)}{\partial \mathbf{r}_j}\frac{\partial \mathbf{r}_j}{\partial t}\right) \\ &= \frac{P_i}{\rho_i^2}\sum_j m_j \mathbf{v}_{ij} \cdot \frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij}, h)}{\partial \mathbf{r}_i},\end{aligned} \tag{56}$$

using the fact that $\mathbf{v}_i = \frac{d\mathbf{r}_i}{dt}$ and

$$\frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij}, h)}{\partial \mathbf{r}_i} = -\frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij}, h)}{\partial \mathbf{r}_j}. \tag{57}$$

Eq. (56) is the discretized SPH version of Eq. (29), i.e. the conservation of energy equation. Knowing how the internal energies of our particles evolve with time, all we need is our equation of state to determine how our particle pressures will evolve.

2.2.4 CONSERVATION PROPERTIES

Eq. (52) conserves linear momentum, since

$$\begin{aligned}\frac{d}{dt}\sum_i m_i\mathbf{v}_i &= \sum_i m_i\frac{d\mathbf{v}_i}{dt}\\ &= -\sum_i\sum_j m_i m_j\left(\frac{P_i}{\rho_i^2}+\frac{P_j}{\rho_j^2}\right)\frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij},h)}{\partial\mathbf{r}_i} - \sum_{i\neq j}\sum_{j\neq i}\frac{Gm_j}{|\mathbf{r}_{ij}|^3}\mathbf{r}_{ij}\\ &= 0,\end{aligned} \tag{58}$$

where we have used the fact that both $\partial W_{3\mathrm{D}}(\mathbf{r}_{ij},h)/\partial\mathbf{r}_i$ and $\mathbf{r}_{ij}$ are anti-symmetric under exchange of i and j. Similarly, Eq. (52) conserves angular momentum, since

$$\begin{aligned}\frac{d}{dt}\sum_i \mathbf{r}_i\times m_i\mathbf{v}_i &= \sum_i m_i\left(\mathbf{r}_i\times\frac{d\mathbf{v}_i}{dt}\right)\\ &= -\sum_i\sum_j m_i m_j\left(\frac{P_i}{\rho_i^2}+\frac{P_j}{\rho_j^2}\right)\left[\mathbf{r}_i\times\hat{\mathbf{r}}_{ij}\frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij},h)}{\partial r_{ij}}\right] - \sum_{i\neq j}\sum_{j\neq i}\frac{Gm_j}{|\mathbf{r}_{ij}|^3}(\mathbf{r}_i\times\mathbf{r}_{ij})\\ &= 0,\end{aligned} \tag{59}$$

where we have used the fact that

$$\frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij},h)}{\partial\mathbf{r}_i} = \frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij},h)}{\partial\mathbf{r}_{ij}}, \tag{60}$$

$\mathbf{v}_i\times\mathbf{v}_i=0$, and also the antisymmetry of the cross product, i.e. $(\mathbf{r}_a\times\mathbf{r}_b)=-(\mathbf{r}_b\times\mathbf{r}_a)$. Energy is also conserved by the equations, namely Eq. (56). We can see this by noting that the total energy of our fluid is given by

$$E = \sum_i m_i\left[\frac{1}{2}|\mathbf{v}_i|^2+\Phi_i+e_i\right]. \tag{61}$$

Taking the time-derivative of this, we find

$$\frac{dE}{dt} = \sum_i m_i\left(\mathbf{v}_i\cdot\frac{d\mathbf{v}_i}{dt}+\frac{d\Phi_i}{dt}+\frac{de_i}{dt}\right). \tag{62}$$

We already know de_i/dt, so let us find expressions for the other two terms. First,

$$\mathbf{v}_i \cdot \frac{d\mathbf{v}_i}{dt} = -\sum_j m_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2}\right) \mathbf{v}_i \cdot \frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij}, h)}{\partial \mathbf{r}_i} - \sum_{j \neq i} \frac{G m_j}{|\mathbf{r}_{ij}|^3} \mathbf{v}_i \cdot \mathbf{r}_{ij}. \tag{63}$$

Note that the $(P_i/\rho_i^2)\mathbf{v_i}$ term here cancels with a term from the de_i/dt equation. The other term from Eq. (62) is

$$\begin{aligned} \frac{d\Phi_i}{dt} &= -\sum_{j \neq i} G m_j \left[\frac{\partial}{\partial \mathbf{r}_i}\left(\frac{1}{|\mathbf{r}_{ij}|}\right)\frac{\partial \mathbf{r}_i}{\partial t}\right] \\ &= \sum_{j \neq i} \frac{G m_j}{|\mathbf{r}_{ij}|^3} \mathbf{v}_i \cdot \mathbf{r}_{ij}. \end{aligned} \tag{64}$$

Notice that the gravitational energy terms from Eqs. (63) and (64) cancel. Thus, the time derivative of the total energy becomes

$$\frac{dE}{dt} = -\sum_i \sum_j m_i m_j \left[\frac{P_j}{\rho_j^2}\mathbf{v}_i \cdot \frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij}, h)}{\partial \mathbf{r}_i} + \frac{P_i}{\rho_i^2}\mathbf{v}_j \cdot \frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij}, h)}{\partial \mathbf{r}_i}\right] = 0, \tag{65}$$

where once again we have used the fact that the double sum is anti-symmetric with respect to particle indices.

2.3 SPHERICALLY SYMMETRIC SPH

As we will ultimately be simulating the evolution of a spherically symmetric halo, we need to reformulate the equations of SPH in spherically symmetric form. This amounts to averaging the formulas we previously derived over solid angle Ω, turning our SPH particles into spherical shells. Therefore, our DM halo will be discretized into a set of concentric shells. Similarly to the 3D case, each shell is labelled by an index i and characterized by four quantities: its position r_i, its velocity $v_i = dr_i/dt$, its specific internal energy e_i, and its mass m_i. It is important to note how, under the assumption of spherical symmetry, the shell positions and velocities are no longer vectors (as they were in the 3D case), but rather scalar functions.

To begin our derivation of the spherically symmetric version of SPH, we start by recalling our arbitrary function $f(\mathbf{r})$ from Eq. (39). If we assume spherical symmetry, then $f(\mathbf{r})$ is really only a function of $|\mathbf{r}|$, i.e. $f(\mathbf{r}) = f(r)$. Spherically averaging Eq. (39), we obtain

$$f(r) = \frac{1}{4\pi} \int d\Omega \sum_j \frac{m_j}{\rho_j} f_j W_{3\mathrm{D}}(\mathbf{r} - \mathbf{r}_j, h). \tag{66}$$

The kernel is the only term here that depends on angle. Thus, we define our 1D spherically symmetric version of the kernel

$$W(r, r', h) = \frac{1}{4\pi} \int d\Omega \, W_{3\mathrm{D}}(\mathbf{r} - \mathbf{r}', h). \tag{67}$$

If we perform the change of variable

$$\mathbf{q} = \frac{\mathbf{r} - \mathbf{r}'}{h} \longrightarrow \; q = \frac{|\mathbf{r} - \mathbf{r}'|}{h} = \frac{\sqrt{r^2 + r'^2 - 2rr' \cos\theta}}{h} \longrightarrow \; \sin\theta d\theta = \frac{h^2}{rr'} q dq, \tag{68}$$

where θ is the angle between $\mathbf{r}$ and $\mathbf{r}'$, then Eq. (67) becomes

$$W(r, r', h) = \frac{h^2}{2rr'} \int_{|r-r'|/h}^{(r+r')/h} dq \; q W_{3\mathrm{D}}(q). \tag{69}$$

Using this formula, we can calculate the spherically symmetric version of any choice of 3D kernel (Gaussian, cubic spline, etc.). Note that plugging Eq. (67) into Eq. (37) yields

$$f(r) = 4\pi \int dr' \; r'^2 \; f(r') \; W(r, r', h). \tag{70}$$

Using the spherically symmetric version of the conservation of mass equation,

$$dm' = 4\pi r'^2 \rho(r') dr', \tag{71}$$

we can express Eq. (70) as

$$f(r) = \int \frac{dm'}{\rho(r')} \; f(r') W(r, r', h). \tag{72}$$

Finally, using Eq. (38), we can write

$$f(r) = \sum_j \frac{m_j}{\rho_j} f_j W(r, r_j, h). \tag{73}$$

Following the derivation for the 3D case, next, we write the Lagrangian for our 1D spherically symmetric setup

$$L = \sum_j m_j \left[\frac{1}{2} v_j^2 - \Phi_j - e_j \right]. \tag{74}$$

Plugging this into the spherically symmetric version of the Euler-Lagrange equations, i.e.

$$\frac{\partial L}{\partial r_i} = \frac{d}{dt}\left(\frac{\partial L}{\partial v_i}\right), \tag{75}$$

we obtain

$$m_i g_i - \sum_j m_j \frac{\partial e_j}{\partial r_i} = m_i \frac{dv_i}{dt}, \tag{76}$$

similar to Eq. (45). In our spherically symmetric case, g_i has the form

$$g_i = -\frac{GM_i}{r_i^2}, \tag{77}$$

where M_i is the enclosed mass within r_i and is given by

$$M_i = \sum_j m_j \theta(r_i - r_j). \tag{78}$$

$\theta(x)$ is a slightly modified version of the Heaviside step function, with the form

$$\theta(x) = \begin{cases} 0 & x < 0 \\ \frac{1}{2} & x = 0 \\ 1 & x > 0 \end{cases} \tag{79}$$

The modification of $\theta(0) = \frac{1}{2}$ ensures the correct gravitational force that a shell feels due to itself. To see this, we note that in the case of spherical symmetry, the gravitational potential energy is given by

$$U = -\frac{1}{2}\sum_{i,j} \frac{Gm_i m_j}{\max(r_i, r_j)}. \tag{80}$$

The $i = j$ contribution, i.e. the gravitational potential energy of a shell due to itself, is therefore

$$U_{ii} = -\frac{1}{2}\frac{Gm_i^2}{r_i}. \tag{81}$$

Thus, the gravitational acceleration a shell feels due to itself is

$$g_i = \frac{|\mathbf{F}_{\text{grav},\, i}|}{m_i} = -\frac{1}{m_i}\frac{\partial U_{ii}}{\partial r_i} = -\frac{1}{2}\frac{Gm_i}{r_i^2}. \tag{82}$$

Looking at the second term from the left-hand side of Eq. (76), we need an expression for

$\partial e_j/\partial r_i$. We can simply use the spherically symmetric version of Eq. (49)

$$\frac{\partial e_j}{\partial r_i} = \frac{P_j}{\rho_j^2}\frac{\partial \rho_j}{\partial r_i}. \tag{83}$$

Using Eq. (73), we can obtain an expression for ρ_j

$$\rho_j = \sum_k m_k W(r_j, r_k, h). \tag{84}$$

Taking the derivative with respect to r_i, we get

$$\frac{\partial \rho_j}{\partial r_i} = \sum_k m_k \frac{\partial W(r_j, r_k, h)}{\partial r_i}(\delta_{ij} + \delta_{ik}). \tag{85}$$

Therefore, the second term from Eq. (76) becomes

$$\begin{aligned}\sum_j m_j \frac{\partial e_j}{\partial r_i} &= \sum_j m_j \frac{P_j}{\rho_j^2}\left(\sum_k m_k \frac{\partial W(r_j, r_k, h)}{\partial r_i}(\delta_{ij} + \delta_{ik})\right) \\ &= \sum_k m_k m_i \frac{P_i}{\rho_i^2}\frac{\partial W(r_i, r_k, h)}{\partial r_i} + \sum_j m_i m_j \frac{P_j}{\rho_j^2}\frac{\partial W(r_j, r_i, h)}{\partial r_i}.\end{aligned} \tag{86}$$

Relabelling the dummy index k as j, and taking into account the symmetry of W with respect to r_i and r_j, we obtain

$$\sum_j m_j \frac{\partial e_j}{\partial r_i} = \sum_j m_j m_i \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2}\right)\frac{\partial W(r_i, r_j, h)}{\partial r_i}. \tag{87}$$

Thus, we can finally write Eq. (76) as

$$\frac{dv_i}{dt} = -\sum_j \frac{G m_j}{r_i^2}\theta(r_i - r_j) - \sum_j m_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2}\right)\frac{\partial W(r_i, r_j, h)}{\partial r_i}. \tag{88}$$

To obtain a spherically symmetric version of our energy equation, we plug Eq. (84) into Eq. (55) as follows

$$\begin{aligned}\frac{de_i}{dt} &= \frac{P_i}{\rho_i^2}\frac{d}{dt}\left(\sum_j m_j W(r_i, r_j, h)\right) \\ &= \frac{P_i}{\rho_i^2}\sum_j m_j \left(v_i \frac{\partial W(r_i, r_j, h)}{\partial r_i} + v_j \frac{\partial W(r_i, r_j, h)}{\partial r_j}\right).\end{aligned} \tag{89}$$

Together, Eqs. (88) and (89), along with the definition $v_i = dr_i/dt$, determine the evolution of an isolated spherically symmetric compressible fluid/gas with no interactions (beyond gravity) divided up into concentric shells using an SPH framework where the smoothing length h is a constant. In the following section, we discuss how these equations change if h depends upon which shell we're looking at and various interactions are introduced.

3 BEYOND STANDARD SPH

3.1 VARIABLE SMOOTHING LENGTH

Though we have yet to state this explicitly, looking at the formulae for different smoothing kernels above, it is evident that the smoothing length parameter h determines how rapidly W falls off as a function of shell (or particle, in our 3D case) spacing. Selecting a suitable value for h is thus important to the accuracy and resolution of an SPH time-evolution [2, 57]. Results from many simulations have shown that, for a given system, values of h that are too small lead to a fluid that is too much like a granular collection of particles and quite sensitive to the effects of discretization. Conversely, if the value of h is too large, the resolution of a simulation suffers as the smoothing length sets a limit on the size of features that can be resolved.

In 3D, for a simulation to run smoothly and give accurate results, a particle should receive contributions from $\mathcal{O}(50-100)$ nearest neighbours [2]. One way to ensure this is to introduce a variable smoothing length h_i associated with particle i that obeys the following

$$\frac{4\pi}{3} h_i^3 n_i = const. \tag{90}$$

where n_i is the number density of particles in the neighbourhood of particle i. In words, Eq. (90) simply means that the number of neighbouring particles in a sphere of radius h_i around particle i is constant. Since $n_i = \rho_i / m_i$, we can re-write the above equation as

$$h_i = \eta \left(\frac{m_i}{\rho_i} \right)^{1/3}, \tag{91}$$

where η is a tuneable constant. (A spherically symmetric version of Eq. (91) can be derived though it makes our equations rather messy. Thus, as we shall discuss in Section 5, we slightly modify the equation for h_i.)

If we bring Eq. (91) into the fold, the equations derived in the sections above need to be modified slightly. First, Eq. (84) becomes

$$\rho_i = \sum_j m_j W(r_i, r_j, h_i), \tag{92}$$

where we have changed $h \to h_i$. Since ρ_i depends on h_i, and vice versa, Eqs. (91) and (92) must be solved simultaneously.

To derive a modified version of Eq. (76), we also need the spatial derivative of ρ_j. Since

h_j depends on ρ_j from Eq. (91), which itself is a function r_j, the derivative is written as

$$\frac{\partial \rho_j}{\partial r_i} = \sum_k m_k \left(\frac{\partial W(r_j, r_k, h_j)}{\partial r_i}(\delta_{ij} + \delta_{ik}) + \frac{\partial W(r_j, r_k, h_j)}{\partial h_j}\frac{\partial h_j}{\partial r_i} \right). \tag{93}$$

The final term in Eq. (93) is the modification to our equation from the previous section. To compute the derivative, we use Eq. (91) as follows

$$\frac{\partial h_j}{\partial r_i} = \frac{\partial}{\partial r_i}\left[\eta \left(\frac{m_j}{\rho_j} \right)^{1/3} \right] = \frac{1}{3}\frac{h_j}{\rho_j}\frac{\partial \rho_j}{\partial r_i}. \tag{94}$$

Note that if we plug this result back into Eq. (93), we'll have a $\partial \rho_j / \partial r_i$ term on both sides of our equation. Rearranging and solving for this term, we obtain

$$\frac{\partial \rho_j}{\partial r_i} = \frac{1}{\Omega_j} \sum_k m_k \frac{\partial W(r_j, r_k, h_j)}{\partial r_i}(\delta_{ij} + \delta_{ik}), \tag{95}$$

where Ω_j is a unitless quantity given by

$$\Omega_j = 1 - \frac{1}{3}\frac{h_j}{\rho_j} \sum_k m_k \frac{\partial W(r_j, r_k, h_j)}{\partial h_j}. \tag{96}$$

Plugging this into Eq. (76) and following the same logic as the previous section, we find

$$\frac{dv_i}{dt} = -\sum_j \frac{G m_j}{r_i^2}\theta(r_i - r_j) - \sum_j m_j \left(\frac{P_i}{\Omega_i \rho_i^2}\frac{\partial W(r_i, r_j, h_i)}{\partial r_i} + \frac{P_j}{\Omega_j \rho_j^2}\frac{\partial W(r_j, r_i, h_j)}{\partial r_i} \right). \tag{97}$$

For our energy equation, we also need the time-derivative of ρ_i, which with the modification of a variable smoothing length becomes

$$\frac{d\rho_i}{dt} = \sum_j m_j \left(v_i \frac{\partial W(r_i, r_j, h_i)}{\partial r_i} + v_j \frac{\partial W(r_i, r_j, h_i)}{\partial r_j} + \frac{\partial W(r_i, r_j, h_i)}{\partial h_i}\frac{\partial h_i}{\partial t} \right). \tag{98}$$

Again, the final term in Eq. (98) is the modification to our corresponding equation from the previous section. To calculate the derivative, we once more use Eq. (91) as follows

$$\frac{dh_i}{dt} = \frac{d}{dt}\left[\eta \left(\frac{m_i}{\rho_i} \right)^{1/3} \right] = \frac{1}{3}\frac{h_i}{\rho_i}\frac{d\rho_i}{dt}. \tag{99}$$

As before, if we plug this result back into Eq. (98), we'll have a $d\rho_i/dt$ term on both sides

of our equation. Again rearranging and solving for this term, we obtain

$$\frac{d\rho_i}{dt} = \frac{1}{\Omega_i} \sum_j m_j \left(v_i \frac{\partial W(r_i, r_j, h_i)}{\partial r_i} + v_j \frac{\partial W(r_i, r_j, h_i)}{\partial r_j} \right). \tag{100}$$

Plugging this result into Eq. (55), we find

$$\frac{de_i}{dt} = \frac{P_i}{\Omega_i \rho_i^2} \sum_j m_j \left(v_i \frac{\partial W(r_i, r_j, h_i)}{\partial r_i} + v_j \frac{\partial W(r_i, r_j, h_i)}{\partial r_j} \right). \tag{101}$$

Eqs. (97) and (101) are our modified versions of Eqs. (88) and (89), adjusted to account for the introduction of variable smoothing lengths. One entirely new equation is needed as well: a formula for dh_i/dt, since h_i depends on ρ_i, which varies with time. Plugging our result from Eq. (100) into Eq. (99), we obtain such a formula

$$\frac{dh_i}{dt} = \frac{1}{3} \frac{h_i}{\Omega_i \rho_i} \sum_j m_j \left(v_i \frac{\partial W(r_i, r_j, h_i)}{\partial r_i} + v_j \frac{\partial W(r_i, r_j, h_i)}{\partial r_j} \right). \tag{102}$$

3.2 SIDM THERMAL CONDUCTION

As mentioned in Section 1, if DM particles have the ability to scatter elastically, heat can be transferred between various regions of a DM halo. The effects of this thermal energy transfer can be incorporated into our equations via a modification to Eq. (29), i.e. our conversation of energy Euler equation.

Recall from Section 2.1 our fluid parcel of volume V. If we allow our fluid to be conductive, heat can enter or leave through the surface of our parcel. Mathematically, this thermal energy flux can be expressed via the following surface integral

$$-\int_S \mathbf{F} \cdot d\mathbf{S}, \tag{103}$$

where $\mathbf{F}$ is the local heat flux density. Using the divergence theorem, we can write this as

$$-\int_V (\nabla \cdot \mathbf{F}) dV. \tag{104}$$

Following the same steps and logic as the derivation in Section 2.1, we obtain the following modified version of Eq. (29)

$$\frac{\partial e}{\partial t} + \mathbf{v} \cdot \nabla e = -\frac{P}{\rho} \nabla \cdot \mathbf{v} - \frac{1}{\rho} \nabla \cdot \mathbf{F}. \tag{105}$$

The heat flux density $\mathbf{F}$ obeys Fourier's law, i.e.

$$\mathbf{F} = -\kappa \nabla T, \tag{106}$$

where κ is the thermal conductivity (or heat conduction coefficient) and T is temperature. For a classical (or Maxwell-Boltzmann) ideal gas, the equipartition theorem tells us that

$$e = \frac{1}{\gamma - 1}\frac{k_B T}{m_{\rm DM}} = \frac{3}{2}\frac{k_B T}{m_{\rm DM}}, \tag{107}$$

where $m_{\rm DM}$ is the DM particle mass and we have assumed our gas is monatomic, i.e. $\gamma = \frac{5}{3}$. Thus, we can re-write Eq. (105) as

$$\frac{\partial e}{\partial t} + \mathbf{v} \cdot \nabla e = -\frac{P}{\rho}\nabla \cdot \mathbf{v} + \frac{1}{\rho}\nabla \cdot (\tilde{\kappa} \nabla e), \tag{108}$$

where $\tilde{\kappa}$ is defined as

$$\tilde{\kappa} = \frac{2}{3}\frac{m_{\rm DM}}{k_B}\kappa. \tag{109}$$

All the quantities in Eq. (109) are constants (or inputs to be chosen) except for the thermal conductivity. To obtain a general expression for κ, we look to transport theory, which tells us that for a spherically symmetric conducting fluid, the heat flux through a surface at r is given by

$$F(r) = \frac{F_+(r) + F_-(r)}{2}. \tag{110}$$

In words, Eq. (110) says that the total flux is simply the average of the outward-moving and inward-moving fluxes. Recall that heat flux is nothing more than particle flux multiplied by thermal energy per particle. Thus, we can write Eq. (110) as

$$F(r) = \frac{1}{2}\left[-\left(\frac{n\lambda}{\tau}\right)\frac{3}{2}k_B T_+ + \left(\frac{n\lambda}{\tau}\right)\frac{3}{2}k_B T_-\right], \tag{111}$$

where n is particle number density, λ is mean-free path, τ is time between collisions, and $T_\pm$ is the temperature just above/below the surface at r. The $\frac{3}{2}k_B T_\pm$ factor is the thermal energy per particle and comes from Eq. (107). To lowest order, we can express $T_\pm$ as

$$T_\pm = T(r) \pm b\lambda\frac{\partial T}{\partial r}, \tag{112}$$

where $b = 25\sqrt{\pi}/32$ is an effective impact parameter, computed using Chapman-Enskog

theory [58, 59]. Plugging this expression into Eq. (111), we find

$$F(r) = -\frac{3}{2} b k_B \frac{n\lambda^2}{\tau} \frac{\partial T}{\partial r}. \tag{113}$$

Comparing this formula to the spherically symmetric version of Eq. (106), we see that the thermal conductivity is simply

$$\kappa = \frac{3}{2} b k_B \frac{n\lambda^2}{\tau}. \tag{114}$$

Thus, using Eq. (109), we can write

$$\tilde{\kappa} = \frac{b\rho\lambda^2}{\tau}, \tag{115}$$

where we have used the fact that $nm_{\rm DM} = \rho$. We need expressions for λ and τ. Recall Eq. (111) applies generally to spherically symmetric conducting fluids. For an SIDM halo, λ and τ depend on the relationship between the mean-free path and the so-called gravitational scale height (or Jeans length) H, given by

$$H = \sqrt{\frac{\nu_{\rm 1D}^2}{4\pi G\rho}}, \tag{116}$$

where $\nu_{\rm 1D}$ is the one-dimensional velocity dispersion, related to the specific energy via

$$\nu_{\rm 1D}^2 = \frac{2}{3} e. \tag{117}$$

The quantity H characterizes the length scale over which objects orbit under the influence of gravity [39, 60]. If the mean-free path is significantly smaller than this, i.e. if $\lambda << H$, then a DM particle undergoes multiple collisions per orbit around the halo center. This regime is known as the short mean-free path (SMFP) limit. In the SMFP regime, τ is given by

$$\tau = \frac{\lambda}{\nu_{\rm 1D}}, \tag{118}$$

and λ is given by

$$\lambda = \frac{1}{\rho\sigma_m}, \tag{119}$$

where σ_m is the DM self-scattering cross section per unit mass (often referred to as the DM cross section). Both theory and simulations suggest that to resolve the small-scale structures issues discussed in Section 1, a cross section of order $\sim 1\ cm^2/g$ is required [39, 61–63].

Plugging Eqs. (118) and (119) into Eq. (115), we can write

$$\tilde{\kappa}_{\rm SMFP} = \frac{b\nu_{\rm 1D}}{\sigma_m}. \tag{120}$$

On the other hand, if $\lambda >> H$, a DM particle orbits the halo center multiple times before scattering. This is known as the long mean-free path (LMFP) limit. In this regime, we replace the mean time between collision τ with the thermal relaxation time t_r, given by

$$t_r = \frac{1}{a\rho\nu_{\rm 1D}\sigma_m}, \tag{121}$$

where $a = \sqrt{16/\pi}$ is a coefficient related to elastic sphere scattering for particles that obey a Maxwell-Boltzmann distribution [64]. We also replace $\lambda \rightarrow H$, since in the LMFP limit, the scale height represents the typical distance a particle travels between collisions. Finally, since we are now considering a new regime, we swap $b \rightarrow C$, where $C = 0.75$ is similarly a constant (though it must be determined empirically through comparison to simulation results) [3, 39]. Thus, in this regime, Eq. (115) becomes

$$\tilde{\kappa}_{\rm LMFP} = \frac{C\rho H^2}{t_r} = \frac{aC\sigma_m}{4\pi G}\nu_{\rm 1D}^3\rho. \tag{122}$$

Eqs. (120) and (122) can be combined into a single equation for $\tilde{\kappa}$ as follows

$$\tilde{\kappa} = \left(\tilde{\kappa}_{\rm SMFP}^{-1} + \tilde{\kappa}_{\rm LMFP}^{-1}\right)^{-1} = ab\nu_{\rm 1D}\sigma_m\left[a\sigma_m^2 + \frac{b}{C}\frac{4\pi G}{\rho\nu_{\rm 1D}^2}\right]^{-1}. \tag{123}$$

Now that we have an expression for $\tilde{\kappa}$, we need an SPH version of the conduction term from Eq. (108), i.e. $(1/\rho)\nabla\cdot(\tilde{\kappa}\nabla e)$. To derive this expression, we again consider an arbitrary function $f(\mathbf{r})$. Consider a Taylor-series approximation of $f(\mathbf{r}_j)$ in the vicinity of $f(\mathbf{r}_i)$

$$f(\mathbf{r}_j) - f(\mathbf{r}_i) = \nabla f\Big|_{\mathbf{r}_i}\cdot(\mathbf{r}_j - \mathbf{r}_i) + \frac{1}{2}\frac{\partial^2 f}{\partial r_s \partial r_k}\Big|_{\mathbf{r}_i}(\mathbf{r}_j - \mathbf{r}_i)_s(\mathbf{r}_j - \mathbf{r}_i)_k + \mathcal{O}(\mathbf{r}_j - \mathbf{r}_i)^3. \tag{124}$$

We neglect terms of order three and higher, and multiply Eq. (124) by

$$\frac{\mathbf{r}_{ji}\nabla_i W_{\rm 3D}(\mathbf{r}_{ji}, h_i)}{|\mathbf{r}_{ji}|^2}, \tag{125}$$

where $\nabla_i = \partial/\partial \mathbf{r}_i$. Integrating both sides over $d^3\mathbf{r}_j$, we obtain

$$\int d^3\mathbf{r}_j \frac{f_{ji}}{|\mathbf{r}_{ji}|^2}\mathbf{r}_{ji}\nabla_i W_{3\mathrm{D}}(\mathbf{r}_{ji}, h_i) = \int d^3\mathbf{r}_j \nabla f\Big|_{\mathbf{r}_i} \cdot \mathbf{r}_{ji}\frac{\mathbf{r}_{ji}\nabla_i W_{3\mathrm{D}}(\mathbf{r}_{ji}, h_i)}{|\mathbf{r}_{ji}|^2} + \frac{1}{2}\int d^3\mathbf{r}_j \frac{\partial^2 f}{\partial r_s \partial r_k}\Big|_{\mathbf{r}_i} (\mathbf{r}_{ji})_s(\mathbf{r}_{ji})_k \frac{\mathbf{r}_{ji}\nabla_i W_{3\mathrm{D}}(\mathbf{r}_{ji}, h_i)}{|\mathbf{r}_{ji}|^2}. \quad (126)$$

Recall that $W_{3\mathrm{D}}$ is spherically symmetric and normalized to unity. Given these properties, Refs. [65, 66] show that

$$\int d^3\mathbf{r}_j \; \mathbf{r}_{ji}\frac{\mathbf{r}_{ji}\nabla_i W_{3\mathrm{D}}(\mathbf{r}_{ji}, h_i)}{|\mathbf{r}_{ji}|^2} = 0, \quad (127)$$

and

$$\int d^3\mathbf{r}_j \; (\mathbf{r}_{ji})_s(\mathbf{r}_{ji})_k \frac{\mathbf{r}_{ji}\nabla_i W_{3\mathrm{D}}(\mathbf{r}_{ji}, h_i)}{|\mathbf{r}_{ji}|^2} = \delta_{sk}. \quad (128)$$

Thus, the first integral on the right-hand side of Eq. (126) vanishes, and for the second integral, the terms with off-diagonal elements of the Hessian matrix of f vanish as well. Therefore, Eq. (126) becomes

$$\nabla^2 f\Big|_{\mathbf{r}_i} = 2\int d^3\mathbf{r}_j \frac{f_{ji}}{|\mathbf{r}_{ji}|^2}\mathbf{r}_{ji}\nabla_i W_{3\mathrm{D}}(\mathbf{r}_{ji}, h_i). \quad (129)$$

To obtain an SPH version of Eq. (129), we use Eq. (38), which allows us to write

$$\nabla^2 f\Big|_{\mathbf{r}_i} = 2\sum_j \frac{m_j}{\rho_j}\frac{f_{ji}}{|\mathbf{r}_{ji}|^2}\mathbf{r}_{ji}\nabla_i W_{3\mathrm{D}}(\mathbf{r}_{ji}, h_i). \quad (130)$$

We're just about ready to write an SPH expression for the conduction term from Eq. (108). First, however, we note that

$$\nabla \cdot (\tilde{\kappa}\nabla e) = \tilde{\kappa}\nabla^2 e + \nabla\tilde{\kappa}\cdot\nabla e, \quad (131)$$

and

$$\nabla^2(\tilde{\kappa}e) = \tilde{\kappa}\nabla^2 e + 2\nabla\tilde{\kappa}\cdot\nabla e + e\nabla^2\tilde{\kappa}. \quad (132)$$

Rearranging Eq. (132) to solve for $\nabla\tilde{\kappa}\cdot\nabla\epsilon$ and plugging this result into Eq. (131), we find

$$\nabla\cdot(\tilde{\kappa}\nabla e) = \frac{1}{2}\Big[\nabla^2(\tilde{\kappa}e) - e\nabla^2\tilde{\kappa} + \tilde{\kappa}\nabla^2 e\Big]. \quad (133)$$

Using this result, we can write a discretized version of the conduction term from Eq. (108)

$$\begin{aligned}\left[\frac{1}{\rho}\nabla\cdot(\tilde{\kappa}\nabla e)\right]\Big|_{\mathbf{r}_i} &= \sum_j \frac{m_j}{\rho_i\rho_j}\frac{\mathbf{r}_{ji}}{|\mathbf{r}_{ji}|^2}\nabla_i W_{3\mathrm{D}}(\mathbf{r}_{ji},h_i)\Big[(\tilde{\kappa}_j e_j-\tilde{\kappa}_i e_i)-e_i(\tilde{\kappa}_j-\tilde{\kappa}_i)+\tilde{\kappa}_i(e_j-e_i)\Big]\\ &= -\sum_j \frac{m_j}{\rho_i\rho_j}\frac{(\tilde{\kappa}_j+\tilde{\kappa}_i)(e_j-e_i)}{|\mathbf{r}_{ij}|^2}\mathbf{r}_{ij}\cdot\frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij},h_i)}{\partial\mathbf{r}_i}.\end{aligned} \tag{134}$$

Eq. (134) gives us an SPH expression for our conduction term in Cartesian coordinates. To convert this formula to a spherically symmetric form, we average it over solid angle Ω. This averaging only affects terms that depend on angle, so all we need to compute is

$$\frac{1}{4\pi}\int d\Omega\ \frac{\mathbf{r}_{ij}}{|\mathbf{r}_{ij}|^2}\cdot\frac{\partial W_{3\mathrm{D}}(\mathbf{r}_i-\mathbf{r}_j,h_i)}{\partial\mathbf{r}_i}. \tag{135}$$

Using a modified version of Eq. (68), where $\mathbf{r}\to\mathbf{r}_i$, $\mathbf{r}'\to\mathbf{r}_j$, and $h\to h_i$, we can express Eq. (135) as

$$\frac{1}{2r_ir_j}\int_{|r_i-r_j|/h_i}^{(r_i+r_j)/h_i} dq\ \frac{\mathbf{q}}{q}\cdot\frac{\partial W_{3\mathrm{D}}(q)}{\partial\mathbf{q}}, \tag{136}$$

using the fact that

$$\frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij},h)}{\partial\mathbf{r}_i}=\frac{\partial W_{3\mathrm{D}}(\mathbf{r}_{ij},h)}{\partial\mathbf{q}}\frac{\partial\mathbf{q}}{\partial\mathbf{r}_i}=\frac{1}{h_i}\frac{\partial W_{3\mathrm{D}}(q)}{\partial\mathbf{q}}. \tag{137}$$

Also, notice that

$$\frac{\mathbf{q}}{q}\cdot\frac{\partial W_{3\mathrm{D}}(q)}{\partial\mathbf{q}}=\hat{\mathbf{q}}\cdot\left(\hat{\mathbf{q}}\frac{\partial W_{3\mathrm{D}}(q)}{\partial q}\right)=\frac{\partial W_{3\mathrm{D}}(q)}{\partial q}. \tag{138}$$

This allows to express Eq. (136) as

$$\frac{1}{2r_ir_j}\int_{|r_i-r_j|/h_i}^{(r_i+r_j)/h_i} dq\ \frac{\partial W_{3\mathrm{D}}(q)}{\partial q}=\frac{1}{2r_ir_j}\left[W_{3\mathrm{D}}\left(\frac{r_i+r_j}{h_i}\right)-W_{3\mathrm{D}}\left(\frac{|r_i-r_j|}{h_i}\right)\right]. \tag{139}$$

Thus, we can finally write the spherically symmetric SPH version of the conduction term as

$$\left[\frac{1}{\rho}\nabla\cdot(\tilde{\kappa}\nabla e)\right]\Big|_{\mathbf{r}_i}=\sum_j\frac{m_j}{\rho_i\rho_j}\frac{(\tilde{\kappa}_j+\tilde{\kappa}_i)(e_j-e_i)}{2r_ir_j}\left[W_{3\mathrm{D}}\left(\frac{r_i+r_j}{h_i}\right)-W_{3\mathrm{D}}\left(\frac{|r_i-r_j|}{h_i}\right)\right] \tag{140}$$

.

3.3 ARTIFICIAL VISCOSITY

An artificial viscosity term is sometimes included in SPH frameworks to deal with discontinuous solutions or "shocks" [1, 67] . However, in our simulations (which we will discuss in more depth in Section 5), we never encountered such behaviour. We did, however, encounter oscillations in our shell trajectories, due to our halo being initialized not quite in hydrostatic equilibrium (a result of discretization). We found that introducing an artificial viscosity term into our equations damped these oscillations and led to smoother and more stable results. The pressure due to this artificial viscosity is commonly expressed as

$$P_{\text{visc}} = -\alpha \rho c_s l (\nabla \cdot \mathbf{v}), \tag{141}$$

where α is a constant, c_s is the sound speed in the fluid, and l is a resolution length scale. A common approach in implementing artificial viscosity into SPH is to modify the conservation of momentum equation as follows

$$\frac{P_i}{\Omega_i \rho_i^2} + \frac{P_j}{\Omega_j \rho_j^2} \rightarrow \left(\frac{P_i}{\Omega_i \rho_i^2} + \frac{1}{2}\Pi \right) + \left(\frac{P_j}{\Omega_j \rho_j^2} + \frac{1}{2}\Pi \right), \tag{142}$$

where Π is a contribution due to the introduction of artificial viscosity. Looking at the form of Eq. (141) and the units of the terms in Eq. (142), we express Π as

$$\Pi = \frac{P_{\text{visc}}}{\rho^2} = -\alpha c_s \frac{h}{\rho} (\nabla \cdot \mathbf{v}), \tag{143}$$

where we have replaced $l \rightarrow h$, the resolution length scale for SPH. For simplicity, let us consider a one-dimensional version of this term, and then extend our findings to 3D. In 1D, Eq. (143) becomes

$$\Pi = -\alpha c_s \frac{h}{\rho} \frac{\partial v}{\partial x}. \tag{144}$$

A Taylor-series approximation of $v(x_j)$ in the neighbourhood of $v(x_i)$ is given by

$$v_j = v_i + \left. \frac{\partial v}{\partial x} \right|_{x_i} (x_j - x_i) + \mathcal{O}(x_j - x_i)^2 \longrightarrow \left. \frac{\partial v}{\partial x} \right|_{x_i} \approx \frac{v_i - v_j}{x_i - x_j}. \tag{145}$$

To avoid possible singularities arising with this term, we replace

$$\frac{1}{x_{ij}} \longrightarrow \frac{x_{ij}}{x_{ij}^2 + \epsilon h^2}, \tag{146}$$

where $\epsilon << 1$ is a softening parameter. Inserting these results into Eq. (144), and replacing particle quantities with their averages (i.e. $h \to \bar{h}_{ij} = \frac{h_i + h_j}{2}$), we find

$$\Pi_{ij} = -\alpha \frac{\bar{c}_{s,ij}}{\bar{\rho}_{ij}} \mu_{ij}, \tag{147}$$

where

$$\mu_{ij} = \frac{\bar{h}_{ij}\, v_{ij}\, x_{ij}}{x_{ij}^2 + \epsilon \bar{h}_{ij}^2}. \tag{148}$$

Extending this result to 3D simply entails replacing $x_{ij} \to r_{ij}$ and $v_{ij} x_{ij} \to \mathbf{v}_{ij} \cdot \mathbf{r}_{ij}$. Lastly, we want to impose the restriction that the artificial viscosity only becomes "active" when two particles are approaching one another. Mathematically, this condition is met if $\mathbf{v}_{ij} \cdot \mathbf{r}_{ij} < 0$. Thus, in 3D, Eqs. (147) and (148) become

$$\Pi_{ij} = \begin{cases} -\alpha \frac{\bar{c}_{s,ij}}{\bar{\rho}_{ij}} \mu_{ij} & \mathbf{v}_{ij} \cdot \mathbf{r}_{ij} < 0 \\ 0 & \text{otherwise} \end{cases}, \tag{149}$$

and

$$\mu_{ij} = \frac{\bar{h}_{ij} \mathbf{v}_{ij} \cdot \mathbf{r}_{ij}}{r_{ij}^2 + \epsilon \bar{h}_{ij}^2}. \tag{150}$$

Simulations implementing Π_{ij} have shown that $\alpha = 1$ and $\epsilon = 0.01$ yield good results [1, 2, 67].

Eq. (149) is the typical SPH expression for artificial viscosity. However, we make a slight modification to the formula in light of efforts to implement this term in our code, which utilizes sparse matrices (we will discuss these shortly) to efficiently carry out summations. The $\bar{\rho}_{ij}$ term in Eq. (149) is problematic when performing these summations. Therefore, we replace it with a geometrically averaged term, which doesn't pose such issues

$$\bar{\rho}_{ij} \to \sqrt{\rho_i \rho_j}. \tag{151}$$

To obtain a spherically symmetric form of Eq. (149), we start by expanding the dot product in μ_{ij} as follows

$$\mathbf{v}_{ij} \cdot \mathbf{r}_{ij} = (\mathbf{v}_i - \mathbf{v}_j) \cdot (\mathbf{r}_i - \mathbf{r}_j) = v_i r_i + v_j r_j - \mathbf{v}_i \cdot \mathbf{r}_j - \mathbf{v}_j \cdot \mathbf{r}_i, \tag{152}$$

where v_i and v_j are the radial components of the velocity vectors, as defined in our spherically symmetric SPH setup. To deal with the remaining two dot products, let us imagine we

oriented our position vectors along the z-axis, such that we can write, for example,

$$\mathbf{v}_i = (v_\perp \cos\phi_i,\ v_\perp \sin\phi_i,\ v_i), \tag{153}$$

where the z-component is just the radial component v_i, and the perpendicular components are defined by a magnitude in the perpendicular direction $v_\perp$ as well as an azimuthal angle ϕ_i. This allows us to write

$$\mathbf{v}_i \cdot \mathbf{r}_j = (v_\perp \cos\phi_i,\ v_\perp \sin\phi_i,\ 0) \cdot \mathbf{r}_j + v_i \hat{\mathbf{r}}_i \cdot \mathbf{r}_j. \tag{154}$$

Spherically averaging this expression over ϕ_i from 0 to 2π, the terms with $v_\perp$ and ϕ_i vanish, and since the final term doesn't depend on ϕ_i, we get

$$\langle \mathbf{v}_i \cdot \mathbf{r}_j \rangle_{\phi_i} = v_i r_j (\hat{\mathbf{r}}_i \cdot \hat{\mathbf{r}}_j) = v_i r_j \cos\theta_{ij}, \tag{155}$$

where we define $\cos\theta_{ij} = \hat{\mathbf{r}}_i \cdot \hat{\mathbf{r}}_j$. Thus, our condition for Π_{ij} to be non-zero becomes

$$v_i r_i + v_j r_j - (v_i r_j + v_j r_i)\cos\theta_{ij} < 0, \tag{156}$$

Rearranging Eq. (156), and noting that $\cos\theta_{ij} < 1$, we find

$$v_i r_i + v_j r_j < (v_i r_j + v_j r_i)\cos\theta_{ij} < (v_i r_j + v_j r_i). \tag{157}$$

Comparing the left and right parts of this inequality, we can write

$$(v_i - v_j)(r_i - r_j) < 0. \tag{158}$$

This is nothing but the 1D version of our condition that two particles must be approaching for Π_{ij} to kick in. Comparing the left and middle parts of Eq. (157), we see that

$$v_i r_i + v_j r_j < (v_i r_j + v_j r_i)\cos\theta_{ij}. \tag{159}$$

There are actually two cases to look at when considering Eq. (157):

- **Case 1**: if the inequality $v_i r_j + v_j r_i < 0$ holds, then Eq. (159) becomes

$$\cos\theta_{ij} < \frac{v_i r_i + v_j r_j}{v_i r_j + v_j r_i}. \tag{160}$$

Now we add and subtract $v_i r_j + v_j r_i$ to the numerator of the right-hand side of Eq.

(160), ultimately obtaining

$$\frac{v_i r_i + v_j r_j}{v_i r_j + v_j r_i} = 1 + \frac{v_i r_i + v_j r_j - v_i r_j - v_j r_i}{v_i r_j + v_j r_i}. \tag{161}$$

The inequality associated with Case 1 tells us the denominator on the right-hand side is negative. Eq. (158) tells us the numerator is negative. Therefore

$$\frac{v_i r_i + v_j r_j}{v_i r_j + v_j r_i} > 1. \tag{162}$$

But $\cos\theta_{ij} < 1$, so our final inequality is

$$-1 < \cos\theta_{ij} < 1. \tag{163}$$

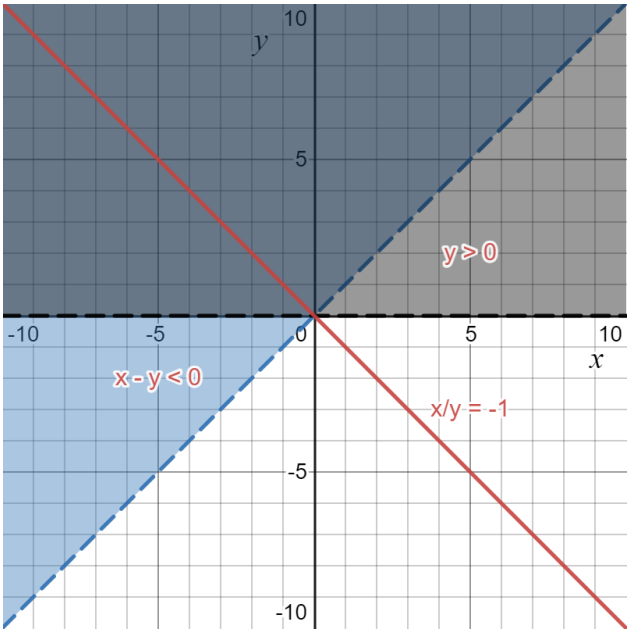


Figure 1: Shaded blue region corresponds to $x - y < 0$. Shaded black region corresponds to $y > 0$. Red line corresponds to $x/y = -1$. (Credit: desmos.com)

- **Case 2**: if the inequality $v_i r_j + v_j r_i > 0$ holds, then Eq. (159) becomes

$$\cos\theta_{ij} > \frac{v_i r_i + v_j r_j}{v_i r_j + v_j r_i}. \tag{164}$$

We must determine whether the right-hand side of Eq. (164) is greater or less than

-1. To accomplish this, let us first define

$$x = v_i r_i + v_j r_j \quad \text{and} \quad y = v_i r_j + v_j r_i. \tag{165}$$

Therefore, we want to determine if $x/y > -1$ or $x/y < -1$, given our two conditions

$$x - y < 0 \quad \text{and} \quad y > 0. \tag{166}$$

Looking at Figure 1, it is apparent that in our region of interest, x/y can be both greater than and less than -1. Thus, our final inequality must be

$$\max(-1, \cos\theta_{\text{crit}}) < \cos\theta_{ij} < 1, \tag{167}$$

where we have defined

$$\cos\theta_{\text{crit}} = \frac{v_i r_i + v_j r_j}{v_i r_j + v_j r_i}. \tag{168}$$

All that remains now to obtain a spherically symmetric form of Eq. (149) is to average Π_{ij} over θ_{ij}. The only term in Π_{ij} that depends of θ_{ij} is μ_{ij}, so we really just need to compute

$$\langle \mu_{ij} \rangle_{\theta_{ij}} \equiv \tilde{\mu}_{ij} = -\frac{1}{2} \int_1^a d\cos\theta_{ij} \, \frac{\bar{h}_{ij} \left[v_i r_i + v_j r_j - (v_i r_j + v_j r_i) \cos\theta_{ij} \right]}{r_{ij}^2 + \epsilon \bar{h}_{ij}^2}, \tag{169}$$

where

$$a = \begin{cases} \max(-1, \cos\theta_{\text{crit}}) & v_i r_j + v_j r_i > 0 \\ -1 & \text{otherwise} \end{cases} \tag{170}$$

Let us define a slightly modified version of Eq. (68)

$$\mathbf{q} = \mathbf{r}_i - \mathbf{r}_j \longrightarrow \; q = |\mathbf{r}_i - \mathbf{r}_j| = \sqrt{r_i^2 + r_j^2 - 2 r_i r_j \cos\theta_{ij}}. \tag{171}$$

It follows that

$$\cos\theta_{ij} = \frac{r_i^2 + r_j^2 - q^2}{2 r_i r_j} \longrightarrow \; \sin\theta_{ij} d\theta_{ij} = \frac{q}{r_i r_j} dq. \tag{172}$$

Therefore, Eq. (169) becomes

$$\tilde{\mu}_{ij} = \frac{1}{2} \int_{|r_i - r_j|}^b dq \, \frac{q}{r_i r_j} \frac{\bar{h}_{ij} \left[v_i r_i + v_j r_j - (v_i r_j + v_j r_i) \left(\frac{r_i^2 + r_j^2 - x^2}{2 r_i r_j} \right) \right]}{q^2 + \epsilon \bar{h}_{ij}^2}, \tag{173}$$

where

$$b = \sqrt{r_i^2 + r_j^2 - 2 r_i r_j a}. \tag{174}$$

Expanding and rearranging Eq. (173), we obtain

$$\tilde{\mu}_{ij} = \frac{\bar{h}_{ij}}{2r_i r_j} \int_{|r_i - r_j|}^{b} dq \left[(v_i r_i + v_j r_j) - \frac{(v_i r_j + v_j r_i)(r_i^2 + r_j^2)}{2r_i r_j} \right] \frac{q}{q^2 + \epsilon \bar{h}_{ij}^2} + \frac{\bar{h}_{ij}}{2r_i r_j} \int_{|r_i - r_j|}^{b} dq \, \frac{(v_i r_j + v_j r_i)}{2r_i r_j} \frac{q^3}{q^2 + \epsilon \bar{h}_{ij}^2}. \quad (175)$$

Using a simple change a variable, one can show that

$$\int_{\alpha}^{\beta} dq \, \frac{q}{q^2 + \epsilon \bar{h}_{ij}^2} = \frac{1}{2} \ln \left(\frac{\beta^2 + \epsilon \bar{h}_{ij}^2}{\alpha^2 + \epsilon \bar{h}_{ij}^2} \right), \quad (176)$$

and also

$$\int_{\alpha}^{\beta} dq \, \frac{q^3}{q^2 + \epsilon \bar{h}_{ij}^2} = \frac{1}{2} \left[\beta^2 - \alpha^2 + \epsilon \bar{h}_{ij}^2 \ln \left(\frac{\alpha^2 + \epsilon \bar{h}_{ij}^2}{\beta^2 + \epsilon \bar{h}_{ij}^2} \right) \right]. \quad (177)$$

Therefore, the first term of Eq. (175) reduces to

$$\tilde{\mu}_{ij}^{(1)} = \frac{\bar{h}_{ij}(r_j^2 - r_i^2)(v_j r_i - v_i r_j)}{8 r_i^2 r_j^2} \ln \left(\frac{b^2 + \epsilon \bar{h}_{ij}^2}{(r_i - r_j)^2 + \epsilon \bar{h}_{ij}^2} \right), \quad (178)$$

and the second term reduces to

$$\tilde{\mu}_{ij}^{(2)} = \frac{\bar{h}_{ij}(v_i r_j + v_j r_i)}{8 r_i^2 r_j^2} \left[b^2 - (r_i - r_j)^2 + \epsilon \bar{h}_{ij}^2 \ln \left(\frac{(r_i - r_j)^2 + \epsilon \bar{h}_{ij}^2}{b^2 + \epsilon \bar{h}_{ij}^2} \right) \right]. \quad (179)$$

Thus, we are finally ready to write our spherically averaged form of Eq. (149)

$$\Pi_{ij} = \begin{cases} -\alpha \frac{\bar{c}_{s,ij}}{\sqrt{\rho_i \rho_j}} \tilde{\mu}_{ij} & (v_i - v_j)(r_i - r_j) < 0 \\ 0 & \text{otherwise} \end{cases} \quad (180)$$

3.3.1 MODIFIED ENERGY EQUATION

The introduction of an artificial viscosity term also effects our energy equation. Recall how for our momentum equation, we implemented Π as indicated in Eq. (142). Following that same approach, our energy formula should be modified as follows

$$\frac{de_i}{dt} = \sum_j m_j \left(\frac{P_i}{\Omega_i \rho_i^2} + \frac{1}{2} \Pi_{ij} \right) \left(v_i \frac{\partial W(r_i, r_j, h_i)}{\partial r_i} + v_j \frac{\partial W(r_i, r_j, h_i)}{\partial r_j} \right). \quad (181)$$

3.3.2 SOUND SPEED

The sound speed associated with shell i is defined as

$$c_{s,i} = \sqrt{\frac{dP_i}{d\rho_i}}. \tag{182}$$

Recall that since entropy is conserved, we have the ideal gas relation $P_i = \zeta \rho_i^{\gamma}$, where ζ is a proportionality constant. Using this relation, Eq. (182) becomes

$$c_{s,i} = \sqrt{\gamma \zeta \rho_i^{\gamma-1}} = \sqrt{\gamma \zeta \rho_i^{\gamma-1} \left(\frac{\rho_i}{\rho_i}\right)} = \sqrt{\frac{\gamma \zeta \rho_i^{\gamma}}{\rho_i}} = \sqrt{\frac{\gamma P_i}{\rho_i}}. \tag{183}$$

4 SPHerical

In this section, we describe how the spherically symmetric SPH equations derived in Sections 2 and 3 are tied together in `SPHerical`, a Python package we created for simulating the evolution of isolated DM halos.

4.1 INITIALIZING A HALO

Initializing a DM halo with `SPHerical` is as simple as running the following piece of code (or something along these lines):

```
import SPHerical.particles as SPH
from SPHerical.analytic_profiles import density
prof = 'NFW'
rho_s = 1.9e+7 # M_sun/kpc^3
rs = 2.59 # kpc
rmin, rmax = 1e-2*rs, 1e+2*rs
density_prof = density(prof,rho_s,rs)
sigma_m = 5 # cm^2/g
N = 100
halo = SPH.particles(density_prof, rmin, rmax, num=N, int_type = 'SIDM',
    int_params = sigma_m)
```

First, we import the `particles.py` module of our package. This file contains code to create an instance of a `particles` class. These objects are the defining feature of the entire `SPHerical` package and store data for all quantities of interest of a halo.

Next, we import the `density` function from the `analytic_profiles` module. This module contains functions that can be used to generate density distributions for several well-known DM profiles (e.g. NFW, Hernquist, pseudo-isothermal). The `density` function allows us to choose which of these profile types will be used to initialize the density distribution of our halo, and what parameters the chosen profile will have. Note: arbitrary density functions can be defined and used in lieu of those in `analytic_profiles`.

After selecting a profile type, we choose values for its parameters. Note the units used here. `SPHerical` assumes that all masses are in dimensions of $M_\odot$ and all lengths are in *kpc*. Notice, however, that the cross section per unit mass parameter `sigma_m` is inputted with different mass and length units. These will be converted into the proper dimensions by `SPHerical`. There is more to be said about the package's unit conventions, however, we will discuss this later.

Finally, we choose the extent of the halo by selecting values for the variables `rmin` and `rmax`. These are not the initial positions of our innermost and outermost SPH shells, but

will be used to determine both, as well as the initial positions of all other shells.

All that is left to do now is pass these inputs, as well as how many shells we would like to divide up our halo into (i.e. `N`) and the interaction type we desire (`'SIDM'` or `'None'`) into the class instantiator `SPH.particles()`. And voilà! We have now successfully initialized a halo in the form of a `particles` object. But what exactly happens to each of these inputs when they are passed into `SPH.particles()`?

4.1.1 SHELL POSITIONS

Once an instance of a `particles` class has been created, the first thing `SPHerical` does is compute the positions of our DM halo shell edges (not to be confused with the shell positions, though closely related). We can pass three keyword arguments into `SPH.particles()` to select how the shell edges will be spaced

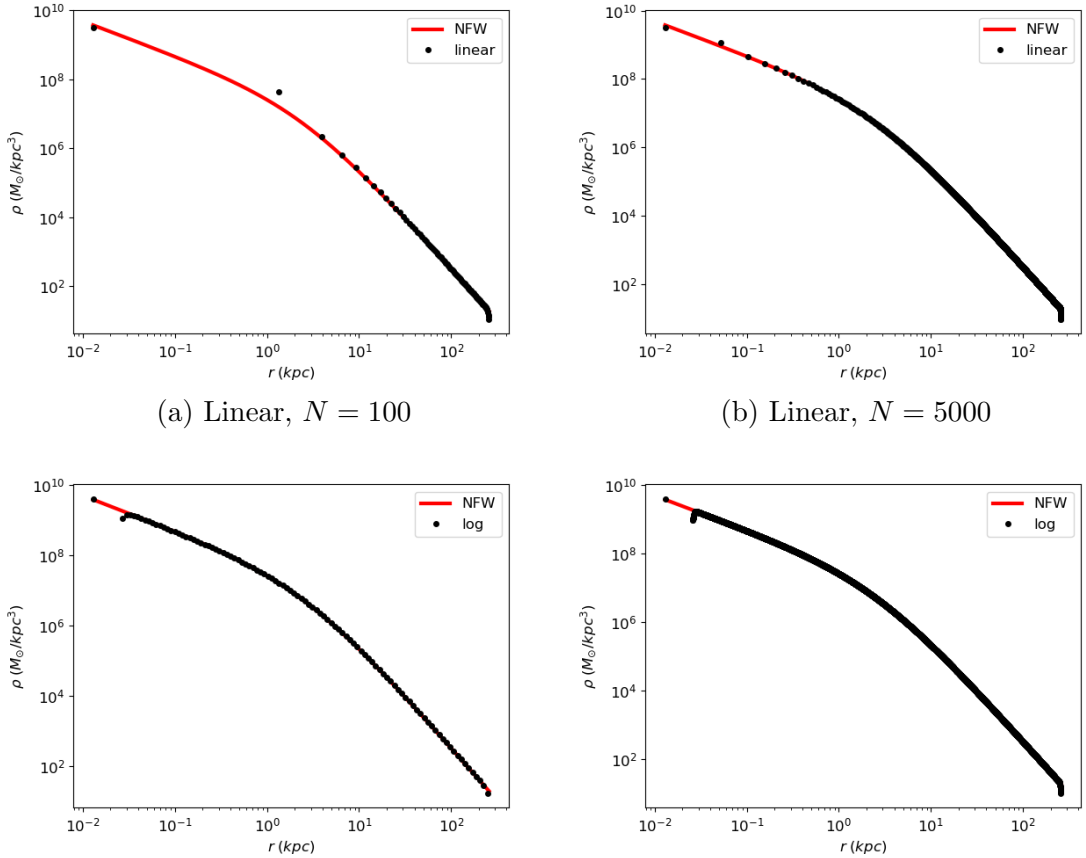
```
halo1 = SPH.particles(..., shell_distribution='linear')
halo2 = SPH.particles(..., shell_distribution='log')
halo3 = SPH.particles(..., shell_distribution='lin_rec')
```

As you might imagine, the `'linear'`/`'log'` inputs result in linearly/logarithmically distributed shell edges, with the inner edge of the innermost shell located at $s_0 = 0$, $s_1 =$ `rmin`, and the outer edge of the outermost shell at $s_{N-1} =$ `rmax`. As you can see in Figure 2a, the linearly distributed shells do a good job of matching SPH densities to the input profile in the outer region of the halo, but fail to do so well in the inner regions. Increasing N to 5000 yields Figure 2b, which displays much better agreement in the inner region, but requires 50 times as many shells. This improvement in accuracy comes at the expense of a significant increase in simulation time, as the computational cost of our SPH equations with summations scales like $\mathcal{O}(NN_{\text{neigh}})$, where N_{neigh} is the number of neighbours in a shell's smoothing region.

Looking at Figure 2c, the log-distributed shells succeed almost everywhere throughout the halo, except for in the innermost region. Increasing N up to 5000 does nothing to resolve this deviation in the inner region, as can be seen in Figure 2d. Ideally, we would like the accuracy of the log distribution in the outer region, and the accuracy of the $N = 5000$ linear distribution in the inner region, but without having to use so many shells. A best-of-both-worlds approach is to use a linear recurrence relation (also called a linear difference equation) to set the positions of the shell edges. This relation produces a smooth transition from linear-spaced edges at small r to log-spaced edges at large r without the need for many shells. The relation has the form

$$s_{i+1} = \alpha s_i + \beta, \tag{184}$$

where as above, we fix $s_0 = 0$, $s_1 = \texttt{rmin}$, and $s_{N-1} = \texttt{rmax}$.



(a) Linear, $N = 100$ (b) Linear, $N = 5000$

(c) Log, $N = 100$ (d) Log, $N = 5000$

Figure 2: Initial SPH densities for different shell distributions. The input density profile used was an NFW with $\rho_s = 1.9 \times 10^7 M_\odot$, $r_s = 2.59$ kpc. Additional `SPHerical` inputs were $\texttt{rmin} = 10^{-2} r_s$, $\texttt{rmax} = 10^2 r_s$, `kernel='CT'`.

We need to determine α and β such that these conditions are satisfied. The first two give us $\beta = \texttt{rmin}$. Off the bat, the third condition does not tell us anything about α, so we'll need to do some work with Eq. (184). We start by writing

$$s_{i+2} = \alpha s_{i+1} + \beta. \tag{185}$$

Subtracting Eq. (184) from Eq. (185), we find

$$s_{i+2} - s_{i+1} = \alpha(s_{i+1} - s_i). \tag{186}$$

Defining $T_i = s_{i+1} - s_i$, we can write

$$T_{i+1} = \alpha T_i \implies T_i = c\alpha^i \quad \text{or} \quad s_{i+1} - s_i = c\alpha^i, \tag{187}$$

where c is a constant which we can determine by setting $i = 0$, i.e.

$$s_1 - s_0 = \beta \implies c = \beta. \tag{188}$$

Using Eq. (187), we can write

$$\begin{aligned} s_n - s_0 &= (s_n - s_{n-1}) + (s_{n-1} - s_{n-2}) + ... + (s_2 - s_1) + (s_1 - s_0) \\ &= \beta(\alpha^{n-1} + \alpha^{n-2} + ... + \alpha + 1) \\ &= \beta \sum_{k=0}^{k=n-1} \alpha^k \\ &= \beta \frac{\alpha^n - 1}{\alpha - 1}. \end{aligned} \tag{189}$$

Thus, we can finally write

$$s_n = \beta \frac{\alpha^n - 1}{\alpha - 1}. \tag{190}$$

Using our condition that $s_{N-1} = \texttt{rmax}$ and recalling $\beta = \texttt{rmin}$, we find

$$\frac{\texttt{rmax}}{\texttt{rmin}} = \frac{\alpha^{\tilde{N}} - 1}{\alpha - 1}, \tag{191}$$

where $\tilde{N} = N - 1$. It is simple to show that for a sequence governed by Eq. (184), a value of $\alpha = 1$ results in perfectly linearly-spaced terms. Increasing this value will cause the sequence to more rapidly transition from linear-spacing initially to log spacing. To ensure our shell edges gently transition in this fashion, we set α equal to

$$\alpha = 1 + \epsilon, \tag{192}$$

where ϵ is a small number. Therefore, we can write Eq. (191) as

$$\frac{\texttt{rmin}}{\texttt{rmax}} = \frac{\epsilon}{(\epsilon + 1)^{\tilde{N}} - 1}. \tag{193}$$

In the limit $\tilde{N} \to \infty$, this equation becomes

$$\frac{\texttt{rmin}}{\texttt{rmax}} = \frac{\epsilon}{\exp(\epsilon\tilde{N}) - 1} \quad \Longrightarrow \quad \frac{\tilde{N}\,\texttt{rmin}}{\texttt{rmax}} = \frac{\epsilon\tilde{N}}{\exp(\epsilon\tilde{N}) - 1}. \tag{194}$$

Now we define $a = (\tilde{N}\ \texttt{rmin})/\texttt{rmax}$, $x = \epsilon\tilde{N}$, and

$$g(x) = \frac{x}{e^x - 1} \quad \Longrightarrow \quad g(x) = a. \tag{195}$$

If we invert Eq. (195), we find

$$g^{-1}(a) = x = \epsilon\tilde{N} \quad \Longrightarrow \quad \alpha = 1 + \frac{1}{\tilde{N}}\, g^{-1}(a). \tag{196}$$

So now all we have to do is solve for $g^{-1}(a)$, or equivalently, x. Recall from Eq. (195) that

$$a = \frac{x}{e^x - 1}. \tag{197}$$

Rearranging and multiplying through by e^{-a-x}, we find

$$(-a - x)e^{-a-x} = -ae^{-a}. \tag{198}$$

Obviously, $x = 0$ is a solution. However, this yields $\alpha = 1$, which we do not desire. It turns out there are additional solutions to equations of the form of Eq. (198), and they can be found using the Lambert W_k function (where k is an integer that denotes a so-called branch)

$$-a - x = W_k(-ae^{-a}). \tag{199}$$

When dealing with purely real numbers (as we are here), the two relevant branches are W_0 and W_{-1}. It turns out, however, that

$$W_0(-ae^{-a}) = -a \ \text{ for } \ a \geq 1 \quad \text{and} \quad W_{-1}(-ae^{-a}) = -a \ \text{ for } \ a \leq 1. \tag{200}$$

Since we do not want the solution $x = 0$, we choose

$$W_k(-ae^{-a}) = \begin{cases} W_0(-ae^{-a}) & a \leq 1 \\ W_{-1}(-ae^{-a}) & a \geq 1 \end{cases} \tag{201}$$

Thus, we can finally write

$$\alpha = \begin{cases} 1 - \frac{1}{N-1}[a + W_0(-ae^{-a})] & a \leq 1 \\ \\ 1 - \frac{1}{N-1}[a + W_{-1}(-ae^{-a})] & a \geq 1 \end{cases} \tag{202}$$

Figure 3 illustrates how the linear recurrence relation prescription compares with the linear and logarithmic distributions. Using only $N = 100$ shells, `SPHerical` is able to quite accurately match the input profile densities in all regions of the SPH halo.

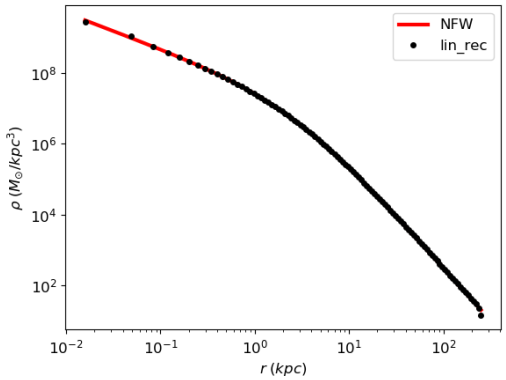


Figure 3: Initial SPH densities for shells distributed according to the linear recurrence relation derived above. `SPHerical` inputs were the same as those in Figure 2.

Once `SPHerical` has determined the positions of the shell edges s_i, the shell positions r_i are simply calculated using the following averaged prescription

$$r_i = \frac{s_{i+1} + s_i}{2}. \tag{203}$$

4.1.2 SHELL MASSES

`SPHerical` initializes the shell masses m_i in a relatively straightforward manner. First, the package determines the mass enclosed interior to each of the shell edges (i.e. within $r = s_i$) by numerically solving the conservation of mass equation for a continuous symmetric sphere

$$\frac{\partial M_{\rm encl}(r)}{\partial r} = 4\pi r^2 \rho_{\rm prof}(r), \tag{204}$$

where $\rho_{\rm prof}(r)$ comes from our input `density_prof` and $M_{\rm encl}(s_0) = 0$. Then, to compute m_i, we simply calculate the mass enclosed between neighbouring shell edges as follows

$$m_i = M_{\rm encl}(s_{i+1}) - M_{\rm encl}(s_i). \tag{205}$$

4.1.3 SHELL DENSITIES AND SMOOTHING LENGTHS

Recall from Section 3 that the equations for the shell densities ρ_i and smoothing lengths h_i must be solved simultaneously. The way in which `SPHerical` goes about accomplishing this is by first approximating the initial densities as follows

$$\rho_i = \rho_{\text{prof}}(r_i). \tag{206}$$

Next, these densities are plugged into Eq. (91) to approximate the initial smoothing lengths h_i. (We will delay discussion of choosing the parameter η until later). Now, Eq. (91) is satisfied. But the equation for ρ_i, i.e. Eq. (92), is not. We can plug our approximate smoothing lengths h_i into this equation to obtain ρ_i that satisfy Eq. (92), but then Eq. (91) is no longer satisfied, and we are back to square one.

`SPHerical` solves this problem by taking the initial estimates for ρ_i and h_i and employing the iterative secant method to simultaneously solve Eqs. (91) and (92). The package uses the built-in `optimize.newton` function imported from the `Scipy` library.

The following is a basic example of how the secant method works: suppose we have a fairly well-behaved function $f(x)$ and that $f(r) = 0$. Also suppose that x_0 and x_1 closely straddle r, and thus $f(x_0)f(x_1) < 0$, i.e. $f(x)$ changes sign over the interval $[x_0, x_1]$. We can thus construct a line through the points $(x_0, f(x_0))$ and $(x_1, f(x_1))$ with equation

$$y = \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_0) + f(x_0). \tag{207}$$

The root of this equation is

$$x_2 = x_0 - f(x_0)\frac{x_1 - x_0}{f(x_1) - f(x_0)}. \tag{208}$$

x_2 is an improved estimate of r relative to x_0 and x_1. To get an even better estimate, we repeat the above steps subject to the following conditions: if $f(x_0)f(x_2) < 0$, then $x_1 \to x_2$. Otherwise, $x_0 \to x_2$. Repeating this process leads to better and better estimates of r.

4.1.4 KERNELS

Recall that the smoothing kernel $W(r, r', h)$ is the defining feature of the entire SPH framework. This kernel is needed to compute quantities of interest associated with SPH shells using some of the equations derived in Sections 2 and 3. Looking at a formula like Eq. (92), a natural way to implement this equation in code is

```
rho = np.zeros(N)
for i in range(N):
    for j in range(N):
        rho[i] += m[j] * W(r[i],r[j],h[i])
```

While this script will produce correct results, it is not nearly as efficient as it could be. For one, recall that the kernels used in SPH possess compact support. The cubic spline described by Eq. (36), for example, falls off to zero when two shells are separated by more than $2h_i$. Thus, the above code will unnecessarily iterate over many shell pairings where `W(r[i],r[j],h[i])` $= 0$. We could account for this by including an `if` statement, but it turns out there is a much more efficient way to perform the computation using sparse matrices.

A sparse matrix is simply a matrix which consists of mostly zero elements. We can represent the kernel for all combinations of shells i and j as the sparse matrix W_{ij}. The library `Scipy` contains a package called `sparse` used for efficiently storing and manipulating sparse matrices. To construct a sparse matrix for the cubic spline kernel, for example, we start by determining which elements of W_{ij} will be non-zero

```
# Define matrix R_diff_ij = r_i - r_j
R_diff = np.subtract.outer(r,r)
# Define matrix H_ij = h_i
H = np.multiply.outer(h,np.ones_like(h))
# Find lists of entries (i,j) where abs(r_i - r_j) < 2*h_i
# rows = [ list of all i values ]
# cols = [ list of all j values ]
rows, cols = np.nonzero(np.abs(R_diff) < 2*H)
```

Next, we assemble our sparse matrix using this information as follows

```
# Evaluate W(r_j,r_i,h_i) at each of these values
data = [ W(r[j],r[i],h[i]) for i,j in zip(rows,cols) ]
W_ij = sparse.coo_matrix((data, (rows, cols)), shape=(num_particles,
    num_particles)).tocsr()
```

Sparse matrices can be constructed in a variety of formats. Coordinate list or COO format, where elements are stored as a tuple `(row, column, value)`, is a fast format when it comes to building a sparse matrix. This format also has the advantage that it can be quickly converted to formats such as compressed sparse row or CSR, which are optimized for manipulation. With our sparse matrix W_{ij} constructed in COO format and converted to CSR, all that is left to do to calculate densities according to Eq. (92) is

```
rho = W_ij.dot(m_list)
```

Figure 4 shows the difference in the time required to compute Eq. (92) using sparse matrices versus the `for` loop approach for a variety of N. The results speak for themselves. If we take $N = 50$, the sparse matrices method is ~ 5.5 times faster than the `for` loop. If $N = 100$, that number is ~ 7.5, and for $N = 200$ it is ~ 9.

Recall that the smoothing kernel appears in a variety of forms throughout our equations, i.e. $\partial W/\partial r_i, \partial W/\partial h_i, W_{3D}$, etc. `SPHerical` creates a sparse matrix for all these kernel terms and appends them as an attribute to our instance of the `particles` class. The package uses matrix multiplication to carry out sums instead of `for` loops, significantly speeding up computation time.

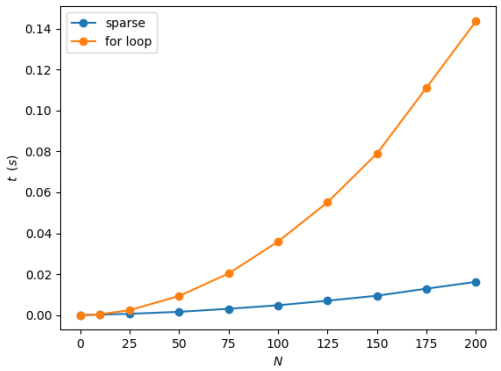


Figure 4: Time required to calculate Eq. (92) using sparse matrices versus the `for` loop method for a variety of N. Each data point is an average over 200 computations.

4.1.5 VELOCITIES, PRESSURES, AND INTERNAL ENERGIES

`SPHerical` allows users to input specific velocity and pressure functions as a keyword argument when creating an instance of the `particles` class. If the user opts for this choice, then the initial velocities v_i and pressures P_i are simply calculated as

```
v_list = [velocity_function(r_i) for r_i in r]
P_list = [pressure_function(r_i) for r_i in r]
```

Otherwise, the halo is initialized in hydrostatic equilibrium. This amounts to setting $v_i = 0$, such that the spherically symmetric version of Eq. (18) becomes

$$\frac{\partial P(r)}{\partial r} = -\frac{GM_{\text{encl}}(r)\rho_{\text{prof}}(r)}{r^2}, \tag{209}$$

where $G = 4.302 \times 10^{-6}\ (kpc/M_\odot) \cdot (km/s)^2$. Note the combination of units used here.

As in Section 4.1.2, `SPHerical` numerically solves Eq. (204). This time, however, an interpolation function is created from the solution to approximate $M_{\rm encl}(r)$. With this interpolating function and the initial condition $P(\infty) = 0$, the package numerically solves Eq. (209) at each shell position r_i, yielding the initial shell pressures P_i. To initialize the shell internal energies, we simply use our equation of state, i.e. Eq. (30).

4.2 TIME EVOLUTION

Once our halo has been initialized, time-evolving it with `SPHerical` is as simple as running the following piece of code (or something similar):

```
import SPHerical.dynamics as dynamics
t_fin = 1 # Gyr
N_steps = 1000
t_steps = np.linspace(0, t_fin, num=N_steps)
dynamics.evolve(halo, t_steps, filename='test_evolution', method='RK45')
```

First, we import the `dynamics` module of `SPHerical`. This module contains all of the code and functions neccessary to time-evolve our halo. Next, we choose the duration we desire for our evolution and at how many time steps we want to save data. We also select the file name we would like our data written to and the ODE solver we want the evolution to be carried out using. We will discuss the options for this solver in more depth shortly.

These inputs are then passed into the `evolve()` function, which works as follows. First, the following list is created

$$y_0 = [r_0, r_1, ..., r_{N-1}, v_0, v_1, ..., v_{N-1}, e_0, e_1, ..., e_{N-1}, h_0, h_1..., h_{N-1}].$$

This is the initial state of our halo. Notice that the quantities in this list are all those that appear of the left-hand side of our SPH equations with time derivatives. All other quantities of interest can be calculated from these (plus m_i).

Next, y_0 is passed into our chosen ODE solver function. Each solver works differently than the others, but they all require a function that returns

$$\frac{dy}{dt} = \left[\frac{dr_0}{dt}, \frac{dr_1}{dt}, ..., \frac{dr_{N-1}}{dt}, \frac{dv_0}{dt}, \frac{dv_1}{dt}, ..., \frac{dv_{N-1}}{dt}, \frac{de_0}{dt}, \frac{de_1}{dt}, ..., \frac{de_{N-1}}{dt}, \frac{dh_0}{dt}, \frac{dh_1}{dt} ..., \frac{dh_{N-1}}{dt}\right].$$

`SPHerical` calculates dy/dt using its `derivatives` module, which contains all the functions needed to compute all the right-hand sides of our equations with time derivatives. Recall from Section 4.1.4 that these sums are all carried out using sparse matrices. For example, to calculate the right-hand side of Eq. (101), `SPHerical` runs the following code

```
# Define diagonal matrix de_drho_over_Omega = P/rho^2/Omega
de_drho_over_Omega = sparse.diags(P_list/rho_list**2/Omega_list, shape=(
    num_particles,num_particles), format='csr')
# Define diagonal matrix V = [..., vi, ...]
V = sparse.diags(v_list, shape=(num_particles,num_particles), format='csr'
    )
de_dt = (de_drho_over_Omega@V@dW_dr_ij + de_drho_over_Omega@dW_drp_ij@V).
    dot(m) * kpc_per_km_times_sec_per_Gyr
```

The `@` symbol in the above code denotes matrix multiplication. Notice that subscript j terms from Eq. (101) are to the right of subscript i terms, as should be the case when performing matrix multiplication. Also, notice this multiplicative term `kpc_per_km_times_sec_per_Gyr`. This is a conversion factor needed to keep the units of all the shell quantities consistent and appears on the right-hand side of all time derivative equations.

With y_0 and dy/dt, our ODE solver is ready to begin time-evolving. As mentioned above, each solver's integration scheme is different. In the following sections, we look at two of the most commonly used solvers in more depth: the leapfrog integrator and the embedded Runge-Kutta integrator.

4.2.1 LEAPFROG INTEGRATOR

If the position, velocity, and acceleration of a particle at time t_n is given by r_n, v_n, and a_n, respectively, then we can write the following Taylor expansions for these quantities at $t_n + \delta t$, i.e. t_{n+1}

$$a_{n+1} = a_n + \delta t\, \dot{a}_n + \mathcal{O}(\delta t^2) \implies \delta t\, \dot{a}_n \approx a_{n+1} - a_n, \tag{210}$$

$$v_{n+1} \approx v_n + \delta t\, a_n + \left(\frac{\delta t}{2}\right) \delta t\, \dot{a}_n \approx v_n + \frac{\delta t}{2}\left(a_n + a_{n+1}\right), \tag{211}$$

$$r_{n+1} \approx r_n + \delta t \left(v_n + \frac{\delta t}{2} a_n\right). \tag{212}$$

Eqs. (211) and (212) together form the leapfrog integrator scheme. This algorithm is both time-reversible and symplectic (i.e. by definition, it conserves energy and angular momentum). There is, however, a problem with trying to use this integrator in its current form to time-evolve our SPH setup. Notice how in Eq. (211), v_{n+1} depends on a_{n+1}. This is fine if the acceleration does not depend on velocity, but looking at our equations for the pressure and viscosity forces, this is clearly not the case. Thus, we must make the following modifications to the integration scheme [1]:

1. First, we predict positions at $t_{n+\frac{1}{2}}$ using

$$r_{n+\frac{1}{2}} = r_n + \frac{\delta t}{2} v_n. \tag{213}$$

2. Next, we use the same logic to determine $v_{n+\frac{1}{2}}$. We also compute the $t_{n+\frac{1}{2}}$ values of our other quantities of interest. For quantities like h, which appear on the left-hand side of time derivatives, this is accomplished as follows

$$h_{n+\frac{1}{2}} = h_n + \frac{\delta t}{2} \left(\frac{dh}{dt} \right)_n. \tag{214}$$

   Once these quantities have been updated, we update ρ, P, Ω, and Π (in that order).

3. Using our updated $t_{n+\frac{1}{2}}$ quantities, we calculate

$$a_{n+\frac{1}{2}} = \left(\frac{dv}{dt} \right)_{n+\frac{1}{2}}. \tag{215}$$

4. Then, we obtain velocities at t_{n+1} using

$$v_{n+1} = v_n + \delta t \; a_{n+\frac{1}{2}}. \tag{216}$$

5. We can obtain positions at t_{n+1} using

$$r_{n+1} = r_n + \delta t \left(\frac{v_n + v_{n+1}}{2} \right). \tag{217}$$

6. To update the remainder of our quantities, we repeat the latter part of step 2, but replace Eq. (214) with

$$h_{n+1} = h_n + \delta t \left(\frac{dh}{dt} \right)_{n+\frac{1}{2}}. \tag{218}$$

7. Repeat steps 1-6 as required.

Using the analytical solution from an ODE, such as one that governs a harmonic oscillator, we can determine the local truncation error in positions (or velocities) at time t_{n+1} associated with this modified leapfrog scheme

$$\epsilon_{n+1} = \left| r_{n+1}^{\text{exact}} - r_{n+1}^{\text{leap}}(r_n^{\text{exact}}, v_n^{\text{exact}}, v_{n+1}^{\text{exact}}) \right|. \tag{219}$$

Computing the average error for a variety of step sizes and number of steps, we can construct the following table

δt	Steps	$\overline{\epsilon}_{n+1}$
0.01	10000	7.0395e-08
0.005	20000	8.7996e-09
0.0025	40000	1.0999e-09
0.001	100000	7.0396e-11

Table 1: Above results were generated using Eq. (219) and an exact ODE solution describing a damped harmonic oscillator.

We make the reasonable assumption that

$$\overline{\epsilon}_{n+1} = c\ \delta t^m, \tag{220}$$

where c is a constant of proportionality. Taking the logarithm of both sides and rearranging, we obtain the following linear equation

$$y = mx + \log(c)\ , \tag{221}$$

where $y = \log(\overline{\epsilon}_{n+1})$ and $x = \log(\delta t)$. Using Eq. (221) and Table 1, along with a linear regression algorithm, we find $m = 2.99999$ with an $r^2 = 0.99999$. In other words, the local truncation error associated with the modified leapfrog integrator goes like δt^3.

Ensuring ϵ_{n+1} is below an acceptable tolerance is important to the accuracy and stability of the leapfrog scheme [1]. Unfortunately, our SPH equations do not have exact solutions, so we cannot use Eq. (219) to determine local truncation errors. We can, however, estimate ϵ_{n+1} as follows.

Consider a $2\delta t$ evolution split up into two steps. First, we time step from $n-1 \to n$, using Eq. (212), but replacing n with $n-1$, i.e.

$$r_n = r_{n-1} + \delta t \left(v_{n-1} + \frac{\delta t}{2} a_{n-1} \right). \tag{222}$$

Next, we time step from $n \to n+1$, using Eq. (212), unmodified. Summing Eqs. (222) and Eq. (212), we find

$$r^a_{n+1} = r_{n-1} + \delta t \left[v_{n-1} + v_n + \frac{\delta t}{2}(a_{n-1} + a_n) \right]. \tag{223}$$

Now, consider the same $2\delta t$ evolution, but with just a single step, i.e.

$$r^b_{n+1} = r_{n-1} + 2\delta t(v_{n-1} + \delta t a_{n-1}). \tag{224}$$

We can construct an error estimate using Eqs. (223) and (224) as follows

$$\epsilon^{\text{est}}_{n+1} = |r^a_{n+1} - r^b_{n+1}|. \tag{225}$$

Following the same steps used above to generate data for Table 1, we obtain

δt	Steps	$\overline{\epsilon}^{\text{est}}_{n+1}$
0.01	10000	8.4477e-07
0.005	20000	1.0559e-07
0.0025	40000	1.3199e-08
0.001	100000	8.4475e-10

Table 2: Above results were generated using Eq. (225) and solving the same damped harmonic oscillator from Table 1.

Using this data along with Eq. (221) and our linear regression algorithm from before, we find $m = 3.000001$ with an $r^2 = 0.99999$, i.e. our error estimate has the same δt^3 dependence as the exact error. Furthermore, we find that, regardless of the choice of step size or values for the damped harmonic oscillator parameters, the ratio $\overline{\epsilon}^{\text{est}}_{n+1}/\overline{\epsilon}_{n+1}$ is always almost exactly 12. This tells us that we should modify Eq. (225) as follows

$$\epsilon^{\text{est}}_{n+1} = \frac{1}{12}\,|r^a_{n+1} - r^b_{n+1}|. \tag{226}$$

To ensure $\epsilon^{\text{est}}_{n+1}$ is not too large at any time step, we impose the condition

$$\epsilon^{\text{est}}_{n+1} < \epsilon^{\text{acc}}_{n+1} = \texttt{atol} + \texttt{rtol}\,|r^{\text{leap}}_{n+1}|, \tag{227}$$

where `atol` is an absolute tolerance, `rtol` is a relative tolerance, and r^{leap}_{n+1} comes from Eq. (217). Both `atol` and `rtol` are keyword arguments of the `evolve()` function, e.g.

```
dynamics.evolve(..., method='leapfrog', rtol=1e-3, atol=1e-6)
```

If Eq. (227) is satisfied for all SPH shells then we accept the time step. Otherwise, we reject it and take a new one. In either case, we can use the fact that $\epsilon_{n+1} \propto \delta t^3$ to determine

a new time step size as follows

$$\delta t^{\text{new}} = f_s \left(\frac{\epsilon_{n+1}^{\text{acc}}}{\epsilon_{n+1}^{\text{est}}} \right)^{1/3} \delta t^{\text{tried}}, \tag{228}$$

where $f_s < 1$ is a safety factor that ensures a conservative choice for δt^{new} [67].

4.2.2 EMBEDDED RUNGE-KUTTA INTEGRATOR

An embedded Runge-Kutta integrator is one ODE integration algorithm of a family of such methods. The particular integrator `SPHerical` employs is known as RK45, where the numbers refer to the order of the method. To get a sense for how RK45 works, we will look at a lower order example [67]. Consider the following general ODE

$$\frac{dy}{dt} = f(t, y). \tag{229}$$

At time t_n, the solution to Eq. (229) is

$$f_n = f(t_n, y_n). \tag{230}$$

At a slightly later time $t_n + \delta t$, we can approximate the solution as

$$f_{n+1} = f(t_{n+1}, y_n + f_n \delta t). \tag{231}$$

At an intermediate time $t_n + \frac{\delta t}{2}$, we can estimate the solution as

$$f_{n+\frac{1}{2}} = f\left(t_{n+\frac{1}{2}}, y_n + \frac{\delta t}{2} \left[\frac{f_n + f_{n+1}}{2} \right] \right). \tag{232}$$

Thus, we can write a second order approximation of y_{n+1} as

$$a = y_n + \delta t \left(\frac{f_n + f_{n+1}}{2} \right). \tag{233}$$

We can also write an improved third order estimate as

$$b = y_n + \delta t \left(\frac{1}{6} f_n + \frac{4}{6} f_{n+\frac{1}{2}} + \frac{1}{6} f_{n+1} \right), \tag{234}$$

where the weights associated with the slopes f are selected to achieve a desired accuracy. (The details are not provided here). Taylor expanding y_{n+1}, we find

$$y_{n+1} = a + c\delta t^3 + \mathcal{O}(\delta t^4) = b + \mathcal{O}(\delta t^4). \tag{235}$$

Thus, to leading order, the local truncation error in our estimate of y_{n+1} is

$$\epsilon_{n+1}^{\text{est}} = |a - b| = c\delta t^3. \tag{236}$$

If $\epsilon_{n+1}^{\text{est}}$ satisfies a version of Eq. (227) modified slightly for RK45, then the time step is accepted and

$$y_{n+1} = b. \tag{237}$$

Otherwise, the time step is rejected. As in the leapfrog method, whether or not this condition is not met, Eq. (228) can be used to determine a new step size.

4.3 ADAPTIVE TIME-STEPPING CRITERIA

The accuracy and stability of both the leapfrog and RK45 methods hinges on the choice of a suitable time step size δt. Both methods have built-in conditions for selecting δt, but there are other criteria to consider as well. In the following sections, we look at the most common time-stepping conditions for fluid dynamics simulations and derive SPH versions of them.

4.3.1 CFL CONDITION

The CFL (Courant–Friedrichs–Lewy) condition in its most general form is given by

$$\delta t_{\text{CFL}} \leq \frac{l}{c}, \tag{238}$$

where l is a characteristic length scale and c is a characteristic velocity scale. In the context of SPH, this condition becomes

$$\delta t_{i,\text{CFL}} \leq \frac{h_i}{c_{s,i}}. \tag{239}$$

This condition ensures that, over the duration of a timestep, information associated with shell i can only propagate to shells within its smoothing region. If artificial viscosity is

included, Eq. (239) must be modified [1, 57, 67]. One form of this modified formula is

$$\delta t_{i,\mathrm{CFL}} = \begin{cases} \frac{0.3h_i}{c_{s,i}+\max_j(\mu_{ij})+1.2\alpha c_{s,i}} & (v_i - v_j)(r_i - r_j) < 0 \\ \\ \frac{0.3h_i}{c_{s,i}} & \text{otherwise} \end{cases}, \tag{240}$$

where 0.3 and 1.2 are numerical factors determined empirically [57, 68]. The inclusion of the term $\max_j(\mu_{ij})$ (which you'll recall is just $h(\nabla \cdot \mathbf{v})$ in 3D) in the denominator of Eq. (240) accounts for fluid expansion/contraction, and thus allows for compressibility effects [1].

4.3.2 FORCE CONDITION

The timescale over which the acceleration of any shell i changes significantly relative to its smoothing region can be expressed as

$$\delta t_{i,\mathrm{F}} = f_{\mathrm{F}}\sqrt{\frac{h_i}{|a_i|}}, \tag{241}$$

where $a_i = dv_i/dt$ and f_{F} is a tuneable parameter, which obeys $f_F < 1$ (similar to the safety factor from 4.2.1). Simulations have shown that f_F in the range $0.25-0.5$ yields good results [1]. Eq. (241) can be used as an upper limit for the time step suitable for evolving shell i.

4.3.3 CHOOSING THE GLOBAL TIMESTEP

While some SPH simulations employ a local timestep, where each particle is time-evolved by its own δt_i, `SPHerical` uses a global timestep, i.e. all shells are evolved in lockstep. At each timestep, this global δt is set as follows

$$\delta t_{\mathrm{glob}} = \min_i(\delta t_{i,\mathrm{CFL}}\ ,\ \delta t_{i,\mathrm{F}}\ ,\ \delta t_i^{\mathrm{new}}), \tag{242}$$

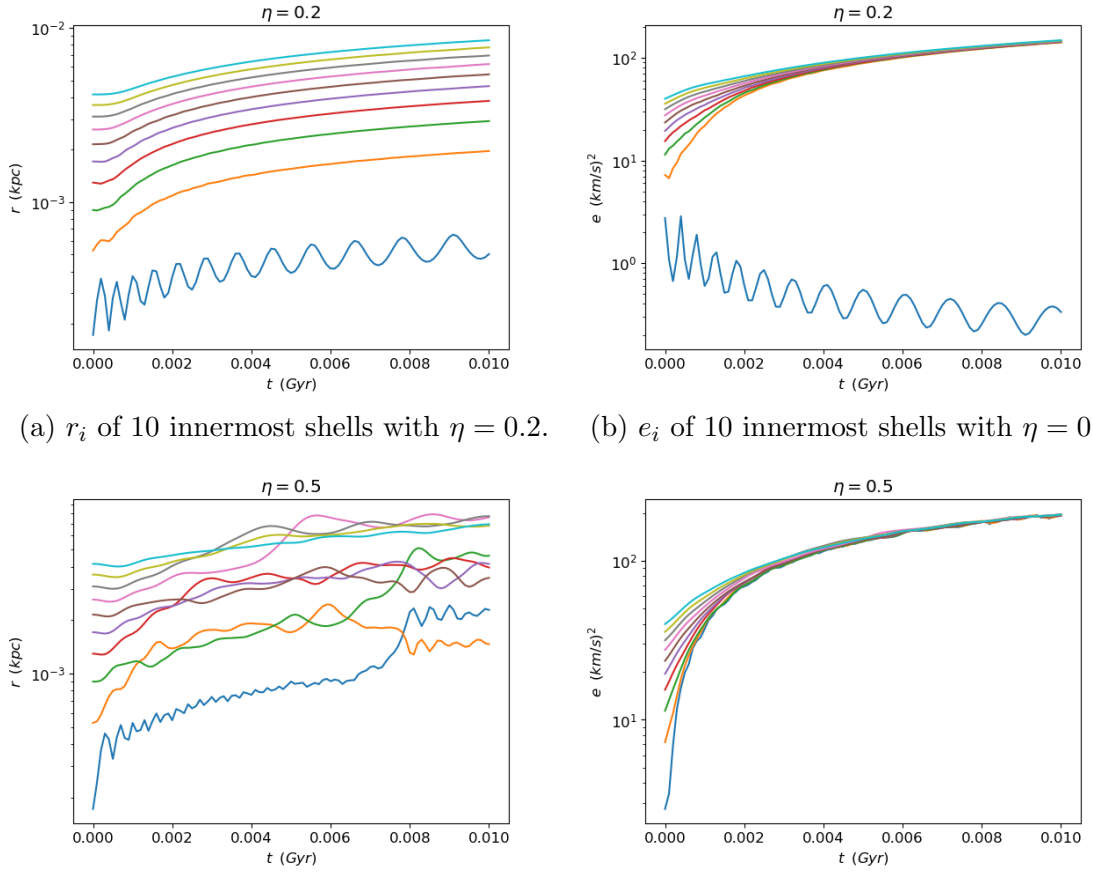
where $\delta t_i^{\mathrm{new}}$ refers to the time step that arises from Eq. (228). The advantage of a global time step is that all particles are on the same footing and there is no information lag. However, individual particle time steps have been shown to significantly reduce overall computation time in certain circumstances. For a more detailed discussion on the topic, see Ref. [1].

5 RESULTS AND CHOOSING PARAMETERS

In this section, we present results generated using `SPHerical` and discuss how these results, as well as preliminary tests, informed our choice of input parameters.

5.1 VARIABLE η

In Section 4.1.3, we stated that we would delay discussion of selecting the η parameter from Eq. (91) until later. The reason for that was because preliminary test time-evolutions we ran using `SPHerical` informed us that the fixed η prescription from our derivation actually yielded poor results.



(a) r_i of 10 innermost shells with $\eta = 0.2$. (b) e_i of 10 innermost shells with $\eta = 0.2$.

(c) r_i of 10 innermost shells with $\eta = 0.5$. (d) e_i of 10 innermost shells with $\eta = 0.5$.

Figure 5: r_i and e_i of the 10 innermost SIDM halo shells over a 0.01 Gyr evolution for low- and high-end values of fixed η. The input density profile used was the same as in Figure 2. Additional `SPHerical` inputs were: $N = 200$, `rmin` $= 10^{-4}\ r_s$, `rmax` $= 10^{2}\ r_s$, `kernel='CT'`, `interaction_type='SIDM'`, `interaction_params` $= 5$.

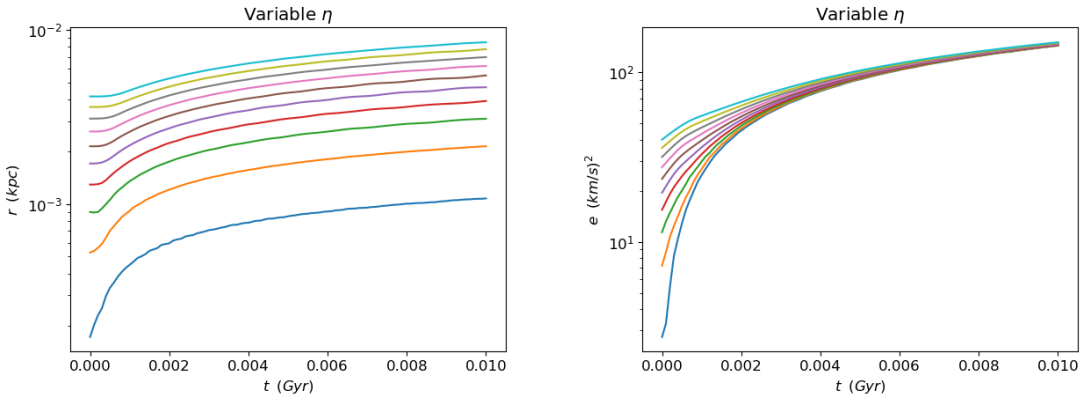
Figure 5 illustrates this well. Looking at panels (a) and (b), we see that if η is small, while the inner shell positions evolve fairly stably, the innermost shell does not "couple" well to its neighbours, and thus, it does not participate properly in energy transfer. On the other hand, if we increase η to a value where the innermost shell does participate, then we start to see shells crossing and merging, a result of pairing instability (Section 5.4 of Ref. [2] has an in-depth discussion on this). See panels (c) and (d).

We explored various avenues to resolve this issue, but ultimately settled upon the following variable η_i prescription

$$\eta_i = \eta_h - (\eta_h - \eta_l)\left(\frac{i}{N-1}\right)^n, \tag{243}$$

where η_h is a high-end value for η_i, η_l is a low-end value, and n is an exponential parameter. All these parameters are tuneable, though empirically, we find $\eta_h = 0.5$, $\eta_l = 0.2$, and $n = 0.05$ to be good values.

The idea behind Eq. (243) is that it allows η_i to transition from a high-end value for inner shells to a lower-end value for more outer shells. This best-of-both-worlds approach allows the innermost shell to properly couple to its neighbours, without introducing any unwanted shell merging/crossing effects (see Figure 6).



(a) r_i of 10 innermost shells with η_i.

(b) e_i of 10 innermost shells with η_i.

Figure 6: r_i and e_i of the 10 innermost shells over a 0.01 Gyr evolution for η_i. Besides the new prescription for η, all other `SPHerical` inputs are the same as in Figure 5.

5.2 CHOOSING THE SMOOTHING KERNEL

Just as with the η parameter, preliminary test simulations using `SPHerical` informed our choice of the smoothing kernel. We initially employed a spherically symmetric version

of the cubic spline kernel given by Eq. (36). Ref [56] has a derivation of the relevant formulae. Using this kernel, however, we obtained results with shell clumping/crossings, as well as oscillatory trajectories (see Figure 9a). These effects were not as a result of any pairing instability (this issue having been handled by introducing our variable η_i) but rather a consequence of tensile instability, i.e. an attractive net force between shells, generally resulting from a negative net pressure [2, 69].

In searching through the literature for solutions to this issue, we came upon Refs. [69] and [70], which propose dealing with this tensile instability by adding to our momentum equation an artificial stress of the form

$$-\sum_j m_j \left(\frac{P_i}{\Omega_i \rho_i^2} \frac{\partial W(r_i, r_j, h_i)}{\partial r_i} + \frac{P_j}{\Omega_j \rho_j^2} \frac{\partial W(r_j, r_i, h_j)}{\partial r_i} \right) \epsilon f_{ij}, \tag{244}$$

where ϵ is a tuneable parameter and

$$f_{ij} = \left[\frac{W_{3\mathrm{D}}(|r_i - r_j|, h_i)}{W_{3\mathrm{D}}(d_i, h_i)} \right]^4, \tag{245}$$

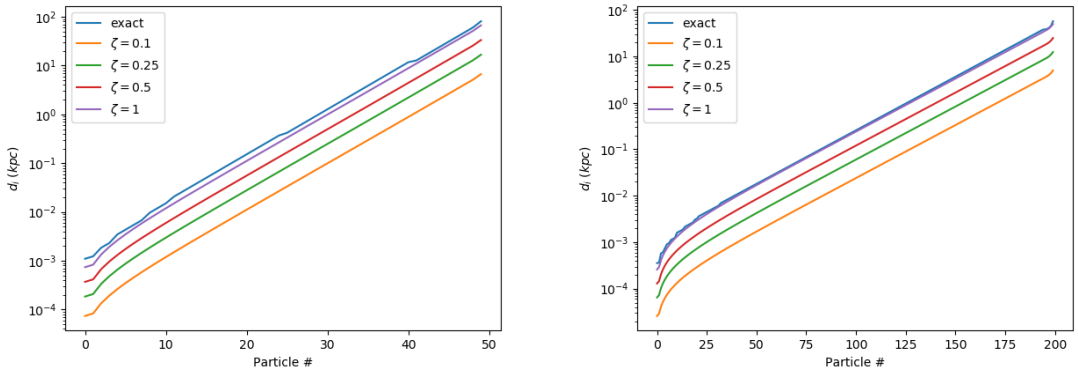
with d_i being the average shell spacing in the smoothing region of shell i. The idea behind including this artificial stress term is to introduce a repulsive force between shells that grows as their separation decreases, thus preventing them from clumping. The term is also constructed in such a way that it is negligible when two shells are not proximate. Note: just as with artificial viscosity, introducing Eq. (244) requires modifying our energy equation accordingly.

Computing d_i in the obvious manner at every time step is computationally expensive, so we determined an alternative method. Starting from the assumption that d_i should be proportional to h_i, we investigated what that proportionality constant ζ should be. Figure 7 shows the results of our investigation. In the limit that N is sufficiently large, it seems d_i and h_i converge, i.e. ζ is unity. Since the spacing between shells does not change drastically as we evolve in time (see Figure 9a), we should simply set $d_i = h_i$.

However, when we do so, we do not obtain the e_i profile we expect (see Figure 10). Instead, we get the profile seen in Figure 8. Looking at this figure, we see that lowering the value of ζ produces better and better results, and that convergence seems to be reached around $\zeta = 0.1$.

Looking at Figure 9, we can see the effect that including this artificial stress term has on the evolution of a `SPHerical` SIDM halo. Even for small values of ϵ, the term yields marked improvement in the shell trajectories, although clumping, crossings, and oscillations

do persist. As we increase ϵ, these effects become less pronounced, and the trajectories become smoother. Figure 9d, however, shows that increasing ϵ too much adversely affects the results in the inner region of the halo.



(a) $N = 50$ (b) $N = 200$

Figure 7: d_i versus particle index calculated exactly and also using the relationship $d_i = \zeta h_i$ for a variety of ζ. Besides the variation in N, all `SPHerical` inputs used were the same as in Figure 5.

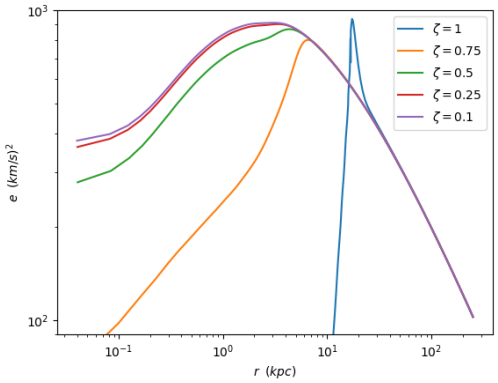


Figure 8: e_i after 0.1 Gyr for various ζ. We use $N = 100$, `rmin` $= 10^{-2}\ r_s$, $\epsilon = 0.1$. Otherwise, all `SPHerical` inputs are the same as in Figure 5.

While including an artificial stress definitely improves the smoothness of shell trajectories, it also has an undesired effect on shell e_i. Looking at Figure 10, we can see that while increasing ϵ smooths out e_i profiles, it also drastically changes them in a manner not in agreement with similar simulations (see Refs. [3] and [63]). At the time of this writing, we do not know why exactly this is the case, though we are actively looking into the issue and hope our investigation will shed some light.

Further searching through the literature led us to Ref. [71], which discusses the inherent tensile instability associated with the cubic spline kernel, and proposes an alternative kernel more stable to clumping: the core triangle (CT) kernel. As with the cubic spline kernel, the CT kernel has compact support. In fact, the two are almost identical except for a small modification that sets the CT apart from spline. We slightly modify the 3D CT kernel given by Ref. [71] to have the following form

$$W_{3\mathrm{D}}(q) = \frac{N}{h_i^3} \begin{cases} (-6a + 9a^2)q + b & 0 < q < 2a \\ 1 - \frac{3}{2}q^2 + \frac{3}{4}q^3 & 2a < q < 1 \\ \frac{1}{4}(2-q)^3 & 1 < q < 2 \\ 0 & \text{otherwise} \end{cases}, \tag{246}$$

where $q = |\mathbf{r}_i - \mathbf{r}_j|/h_i$, $a = 1/3$ and $b = 1 + 6a^2 - 12a^3 = 11/9$ (to ensure that $W_{3\mathrm{D}}, \partial W_{3\mathrm{D}}/\partial r$, and $\partial^2 W_{3\mathrm{D}}/\partial r^2$ are all continuous), and N is a normalization factor given by

$$N = \frac{1}{\pi(6.4a^5 - 16a^6 + 1)}. \tag{247}$$

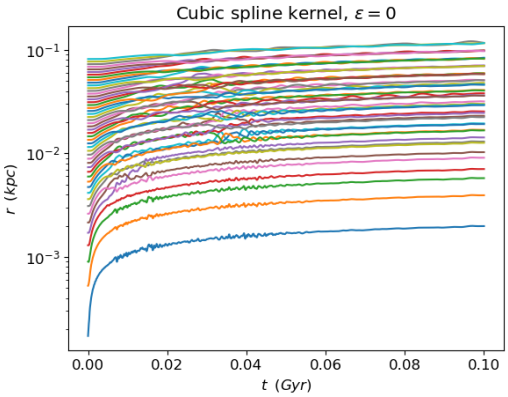
Using Eqs. (67) and (68), we can write

$$W(r_i, r_j, h_i) = \frac{\tilde{N}}{4\pi h_i r_i r_j} \int_{|r_i - r_j|/h_i}^{r_i + r_j/h_i} dq\ 2qW_{3\mathrm{D}}(q), \tag{248}$$

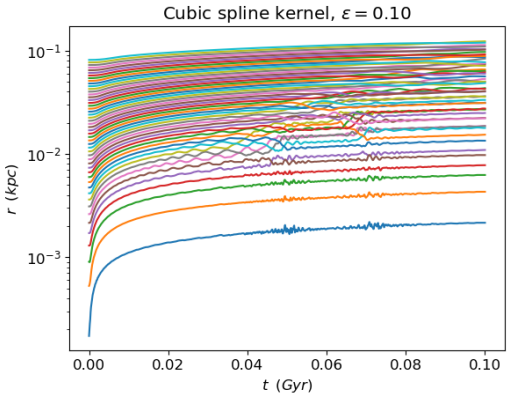
where $\tilde{N} = \pi N$.

Note that

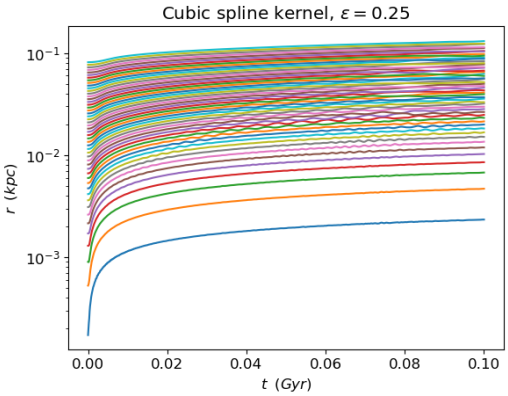
$$\int dq\ 2qW_{3\mathrm{D}}(q) = \begin{cases} A(q) & 0 < q < \frac{2}{3} \\ B(q) & \frac{2}{3} < q < 1 \\ C(q) & 1 < q < 2 \\ 0 & \text{otherwise} \end{cases}, \tag{249}$$



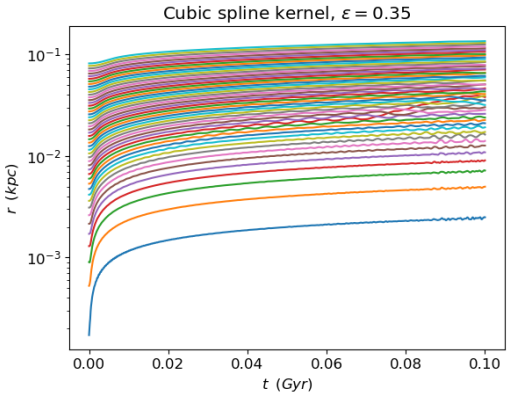
(a) r_i of 50 innermost shells with $\epsilon = 0$.



(b) r_i of 50 innermost shells with $\epsilon = 0.10$.



(c) r_i of 50 innermost shells with $\epsilon = 0.25$.



(d) r_i of 50 innermost shells with $\epsilon = 0.35$.

Figure 9: r_i of the 50 innermost shells over a 0.1 *Gyr* evolution for a variety of ϵ values. Besides for the inclusion of the artificial stress term, all other `SPHerical` inputs are the same as in Figure 5.

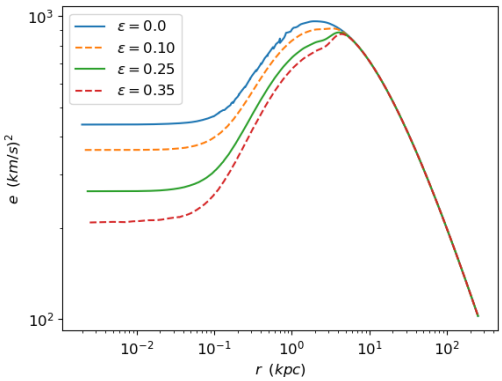


Figure 10: e_i after 0.1 *Gyr* for various ϵ. Besides for the inclusion of artificial stress, all other `SPHerical` inputs are the same as in Figure 5.

where

$$A(q) = -\frac{2}{3}q^3 + \frac{11}{9}q^2, \quad B(q) = q^2 - \frac{3}{4}q^4 + \frac{3}{10}q^5, \quad C(q) = -\frac{1}{10}q^5 + \frac{3}{4}q^4 - 2q^3 + 2q^2. \tag{250}$$

If we now define $\sigma = r_i/h_i$ and $\sigma' = r_j/h_i$, then we can express $W(r_i, r_j, h_i)$ as follows

- If $\sigma + \sigma' < \frac{2}{3}$:

$$W(r_i, r_j, h_i) = \frac{\tilde{N}}{4\pi h_i r_i r_j}\Big[A(\sigma+\sigma) - A(|\sigma-\sigma|)\Big]. \tag{251}$$

- If $\frac{2}{3} < \sigma + \sigma' < 1$:

$$W(r_i, r_j, h_i) = \frac{\tilde{N}}{4\pi h_i r_i r_j}\begin{cases} B(\sigma+\sigma) - A(|\sigma-\sigma|) + \frac{4}{405} & 0 < |\sigma-\sigma| < \frac{2}{3} \\ B(\sigma+\sigma) - B(|\sigma-\sigma|) & \frac{2}{3} < |\sigma-\sigma| < 1 \end{cases} \tag{252}$$

- If $1 < \sigma + \sigma' < 2$:

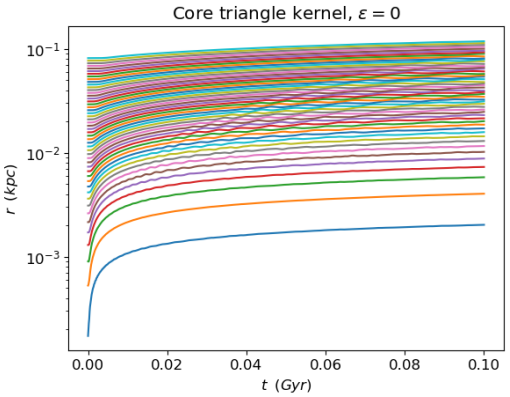
$$W(r_i, r_j, h_i) = \frac{\tilde{N}}{4\pi h_i r_i r_j}\begin{cases} C(\sigma+\sigma) - A(|\sigma-\sigma|) - \frac{73}{810} & 0 < |\sigma-\sigma| < \frac{2}{3} \\ C(\sigma+\sigma) - B(|\sigma-\sigma|) - \frac{1}{10} & \frac{2}{3} < |\sigma-\sigma| < 1 \\ C(\sigma+\sigma) - C(|\sigma-\sigma|) & 1 < |\sigma-\sigma| < 2 \\ 0 & 2 < |\sigma-\sigma| \end{cases} \tag{253}$$

- If $2 < \sigma + \sigma'$:

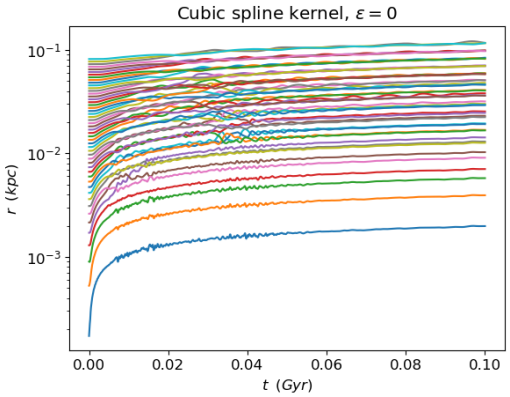
$$W(r_i, r_j, h_i) = \frac{\tilde{N}}{4\pi h_i r_i r_j}\begin{cases} -A(|\sigma-\sigma|) + \frac{115}{162} & 0 < |\sigma-\sigma| < \frac{2}{3} \\ -B(|\sigma-\sigma|) + \frac{7}{10} & \frac{2}{3} < |\sigma-\sigma| < 1 \\ -C(|\sigma-\sigma|) + \frac{8}{10} & 1 < |\sigma-\sigma| < 2 \\ 0 & 2 < |\sigma-\sigma| \end{cases} \tag{254}$$

Figure 11 shows how the CT kernel compares to the cubic spline kernel. Without including the artificial stress from Eq. (244), the difference between the results generated using these two kernels is stark. The CT kernel yields smooth shell trajectories with little

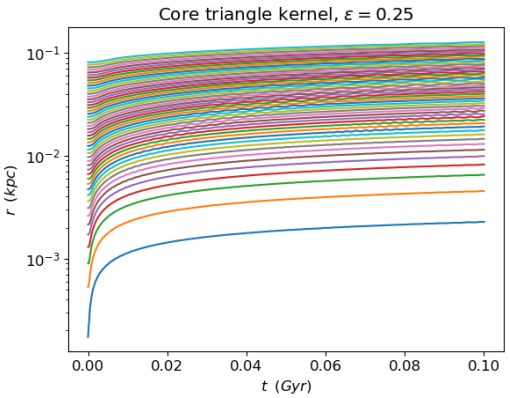
clumping/crossings. With the inclusion of artificial stress, the CT kernel trajectories are still much smoother than the cubic spline, though the difference is not nearly as noticeable. Furthermore, the CT kernel trajectories with artificial stress appear even smoother in some regions than those without.



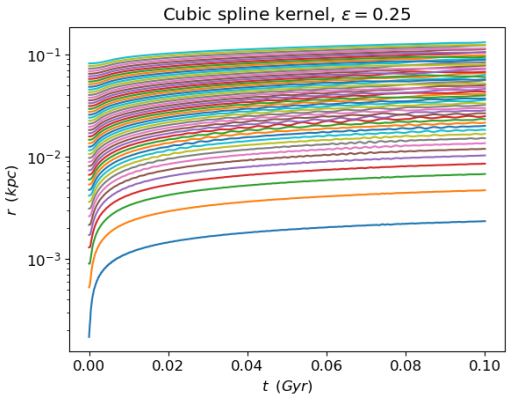
(a) r_i of 50 innermost shells, CT, $\epsilon = 0$.



(b) r_i of 50 innermost shells, spline, $\epsilon = 0$.



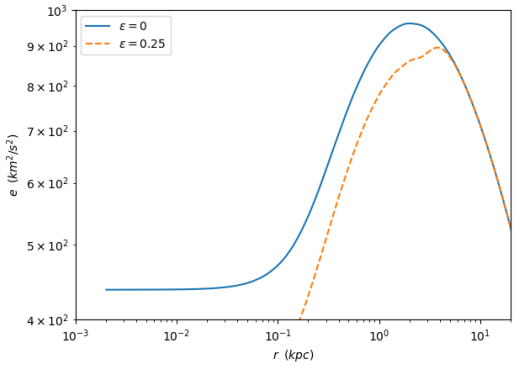
(c) r_i of 50 innermost shells, CT, $\epsilon = 0.25$.



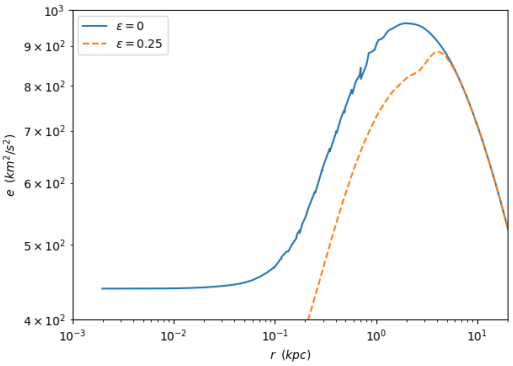
(d) r_i of 50 innermost shells, spline, $\epsilon = 0.25$.

Figure 11: r_i of the 50 innermost shells over a 0.1 Gyr evolution for the CT vs spline kernels. Besides for the inclusion of artificial stress in some runs, all other `SPHerical` inputs are the same as in Figure 5.

However, just as with the cubic spline kernel, the artificial stress yields unwanted effects on shell e_i. Figure 12 illustrates this, as well as the fact that with $\epsilon = 0$, the CT kernel produces a smooth e_i profile without any of the fluctuations that the spline kernel yields.



(a) e_i for CT kernel.



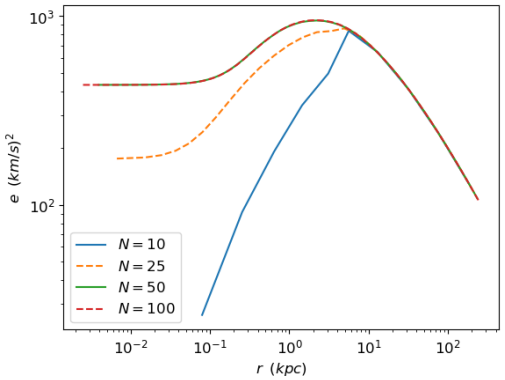
(b) e_i for spline kernel.

Figure 12: e_i after a 0.1 *Gyr* evolution varying ϵ and kernel type. Besides for the inclusion of artificial stress, all other `SPHerical` inputs are the same as those used in Figure 5.

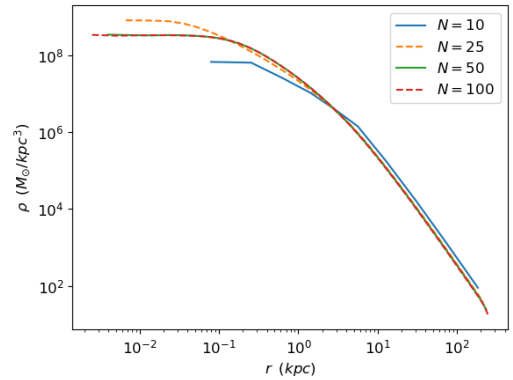
5.3 CHOOSING THE NUMBER OF SHELLS

How many shells is the optimal number for a `SPHerical` simulation? The answer depends on what degree of accuracy one wants in their results and how long they are willing to wait for said results to be generated. That being said, Figure 13 illuminates a great deal about how best to go about choosing N.

$N = 10$ and $N = 25$ seem to be far too few shells, as there is drastic difference between the results generated using them and those produced using higher N. It appears convergence starts to be reached around $N = 50$, and definitely by $N = 100$. Larger values of N produce slightly different results, but with increasingly negligible differences. Also, recall that computation time goes like $\mathcal{O}(NN_{\text{neigh}})$, so increased precision comes at a cost.



(a) e_i after 0.1 *Gyr* evolution.



(b) ρ_i after 0.1 *Gyr* evolution.

Figure 13: Data from 0.1 *Gyr* evolutions using various numbers of shells. Besides the variation in N, all other `SPHerical` inputs are the same as in Figure 5.

5.4 CHOOSING THE INTEGRATOR

In Section 4.2, we introduced the leapfrog and RK45 integrators. In this section, we present the results of tests that we put both integrators through to determine which is faster and more robust when it comes to evolving a `SPHerical` SIDM halo.

5.4.1 CONVERGENCE

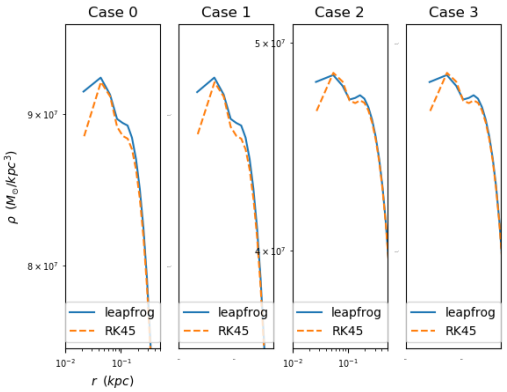
Before testing the speed and other properties of the leapfrog and RK45 integrators (`SPHeric -al` uses the `Scipy` library's `integrate.RK45` function), we wanted to make sure that they gave converging results. Thus, we ran a number of simulations, varying N, σ_m, and `atol`/`rtol` (see Table 3). Figure 14 shows the results. Looking at all of the cases, it is clear that the agreement between the two integrators improves as we increase N, particularly for ρ_i. It is also clear that decreasing the `atol`/`rtol` parameters does not have a significant effect on the results. Further, the discrepancy between leapfrog and RK45 is nowhere greater than $\sim 3.5\%$.

Case	N	σ_m (cm^2/g)	`atol`	$t_{\text{leap}}/t_{\text{RK45}}$	$E_f^{\text{leap}}/E_0^{\text{leap}}$	$E_f^{\text{RK45}}/E_0^{\text{RK45}}$
0	50	5	1.00E-06	1.07609	0.99994	0.99982
1	50	5	1.00E-07	2.04734	0.99994	0.99982
2	50	50	1.00E-06	0.75196	0.99950	0.99946
3	50	50	1.00E-07	1.47501	0.99950	0.99946
4	100	5	1.00E-06	1.82238	1.00015	1.00005
5	100	5	1.00E-07	3.57744	1.00015	1.00004
6	100	50	1.00E-06	0.62182	0.99985	0.99978
7	100	50	1.00E-07	1.06380	0.99985	0.99980
8	200	5	1.00E-06	1.02152	1.00000	0.99995
9	200	5	1.00E-07	1.57774	1.00000	0.99995
10	200	50	1.00E-06	0.47151	0.99963	0.99965
11	200	50	1.00E-07	0.76160	0.99963	0.99965

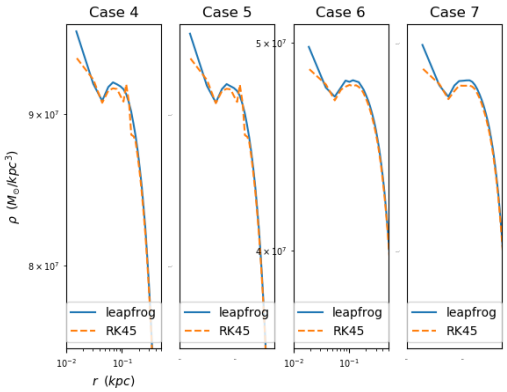
Table 3: Computation time for 1 *Gyr* evolution using RK45 vs leapfrog integrators and varying `SPHerical` input parameters. For each run, `rtol` was taken to be 10^3 `atol`.

5.4.2 COMPUTATION TIME

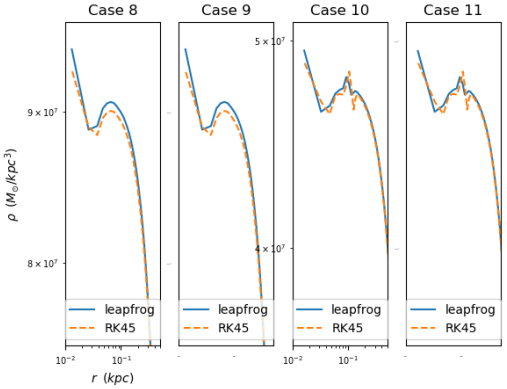
A major factor when selecting an ODE integrator is how long it takes to generate results. To test the speed of the leapfrog and RK45 integrators, we timed all of the runs from the suite of simulations described in the previous section. Table 3 shows these results. For almost every set of input parameters, RK45 outperformed leapfrog in computation time.



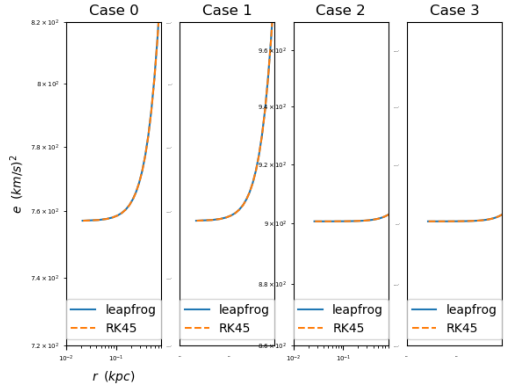
(a) ρ_i for Cases 0-3



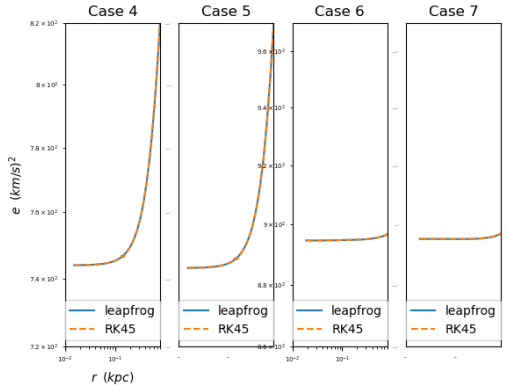
(b) ρ_i for Cases 4-7



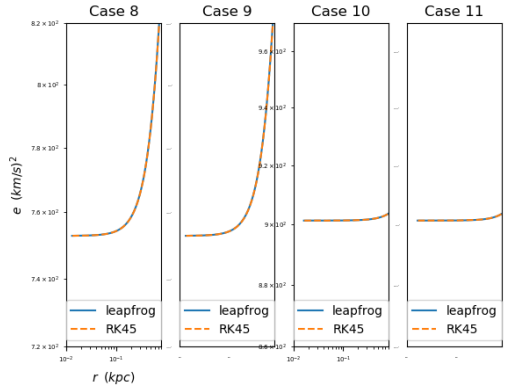
(c) ρ_i for Cases 8-11



(d) e_i for Cases 0-3



(e) e_i for Cases 4-7



(f) e_i for Cases 8-11

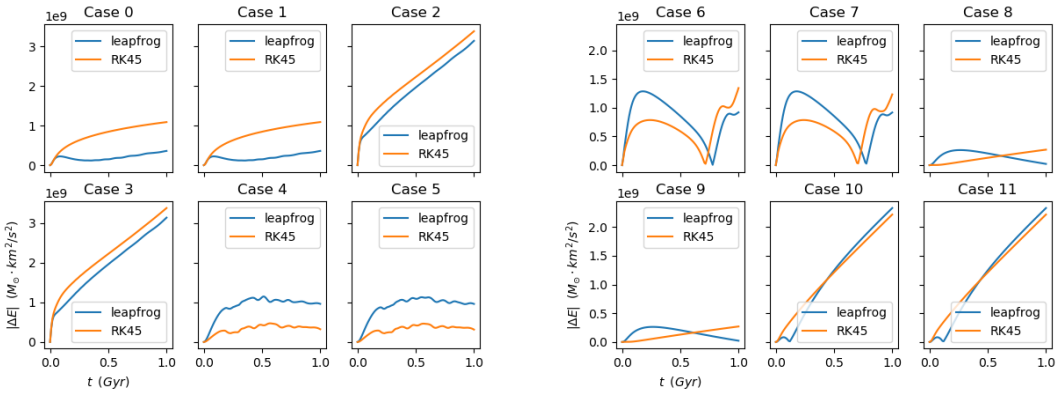
Figure 14: ρ_i and e_i after a 1 Gyr evolution for each of the cases in Table 3. Though not shown, ρ_i and e_i agree to a high degree in the outer part of the halo for all cases.

On average, it was $\sim$ 1.5 times quicker, and in one case, it was over 3.5 times as fast. However, there were a few instances where leapfrog ran in a shorter time, all of them when $\sigma_m = 50\ cm^2/g$. This might be an indication that RK45 struggles with higher cross sections,

however we need to look into this further to say conclusively.

5.4.3 ENERGY CONSERVATION

In addition to computation speed, when dealing with a physical system (such as a DM halo), a major factor in choosing an integrator is how well it conserves energy. Obviously, since we are not solving Euler's equations exactly, we do not expect energy to be perfectly conserved. However, we do expect it to be conserved to a fairly high degree if our integrator is accurate. Using the same suite of tests described in Section 5.4.2, we generated Table 3 and Figure 15 to test and compare the energy conservation capabilities of the leapfrog and RK45 integrators.



(a) $|\Delta E|$ for Cases 0-5

(b) $|\Delta E|$ for Cases 6-11

Figure 15: $|\Delta E|$ over a 1 Gyr evolution for each of the cases in Table 3.

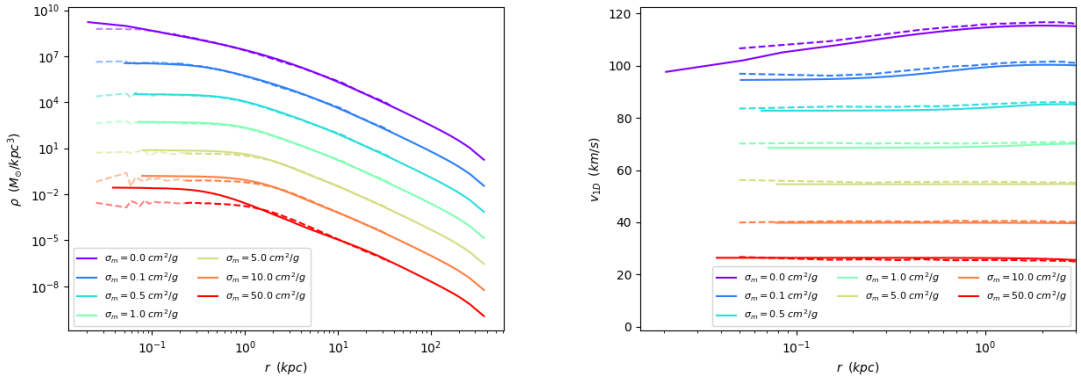
In absolute terms, both conserve energy to a high degree. Cases 0-3 (i.e. all the cases where $N = 50$) indicate a clear advantage in energy conservation for the leapfrog integrator. In Cases 4 and 5, where $N = 100$ and $\sigma_m = 5\ cm^2/g$, RK45 is clearly superior. Over the 1 Gyr evolution period of our simulations, the results of Cases 6-9 are inconclusive, and likely need to be run for a longer duration. Cases 10 and 11 seem to favour RK45 in terms of energy conservation, but also likely should be run for longer than 1 Gyr to yield more conclusive results.

5.5 PROOF OF CONCEPT COMPARISONS

A common practice when testing isolated SIDM simulations like those done using `SPHerical` is to crosscheck their results with cosmological N-body simulations. One such often compared-to simulation set is the "Pippin" runs from Ref. [38]. Pippin refers to a dwarf galaxy

halo whose evolution Ref. [38] simulates with a variety of cross sections, ranging from $\sigma_m = 0\ cm^2/g$ to $\sigma_m = 50\ cm^2/g$. Both Refs. [3] and [63] perform comparisons between their fluid simulations and Pippin using a halo whose initial density profile is modelled as an NFW with parameters fit to the Pippin CDM ($\sigma_m = 0\ cm^2/g$) case and evolving for $\sim 10\ Gyr$.

Following this approach, we generated Figure 16. `SPHerical` does a decent job overall in matching the Pippin results. However, like with Refs. [3] and [63], our simulation overshoots the Pippin densities and undershoots the $v_{1\mathrm{D}}$, likely due to being a different type of simulation. For larger cross sections, the density disagreement is likely due to Pippin being a cosmological simulation, where heating from mergers and infalls can slow down core collapse [22]. Even so, these discrepanices are nowhere greater than a factor of ~ 1.4 for densities and ~ 1.1 for $\nu_{1\mathrm{D}}$.



(a) `SPHerical` vs Pippin densities.

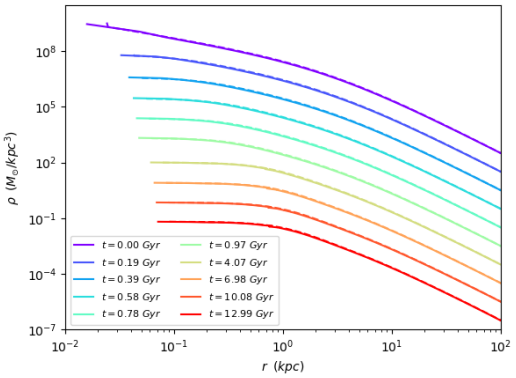
(b) `SPHerical` vs Pippin v_{1D}.

Figure 16: Comparison of `SPHerical` (solid) and Pippin (dashed) results for ρ_i and $\nu_{1\mathrm{D}}$. The density curves are offset by a factor of 50 and the $\nu_{1\mathrm{D}}$ curves by 15 km/s. The semi-transparent dashed lines indicate low resolution results (see Ref. [38]). `SPHerical` inputs: `rmin` $= 10^{-2} r_s$, $N = 100$, `kernel='CT'`, `method='RK45'`. All other inputs are the same as those in Figure 5.

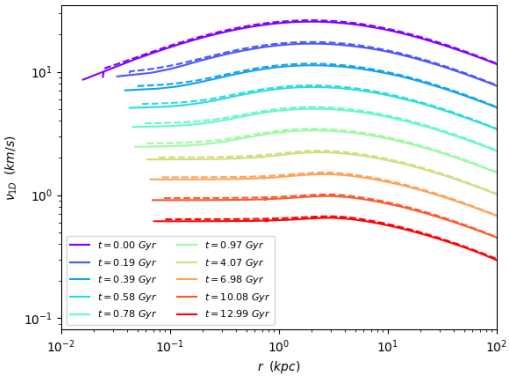
Another comparison that we performed was between `SPHerical` and some results from Ref. [3], which uses a semianalytic gravothermal fluid model similar to ours to simulate an isolated spherical SIDM halo. The crosscheck is in Figure 17. For two choices of σ_m (1 and 10 cm^2/g), our results agree with those of Ref. [3] rather well. Moreover, the results do not simply agree at some final snapshot, but match at a variety of interim times.

We also compared how the central density ρ_c (defined as the density of the innermost shell) of halos of varying cross sections evolved over time. The crosscheck is displayed in Figure 18. The results generated using `SPHerical` and Ref. [3] agree well, although there

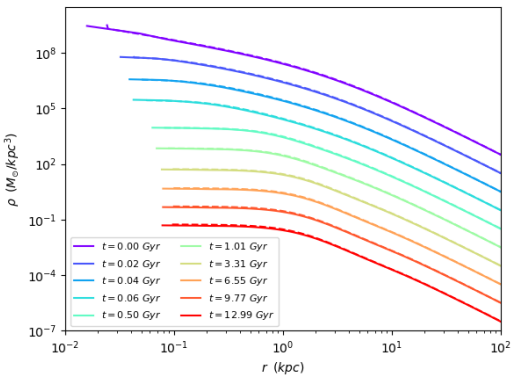
are noticeable deviations. At early times, for $\sigma_m = 0.1\ cm^2/g$, the disagreement is due to the innermost shell density of the Ref. [3] halo being initialized at a slightly larger value than the `SPHerical` halo. At late times, for $\sigma_m = 10.0\ cm^2/g$, we are unsure as to the origin of the discrepancy, though we believe it may simply be a result of the differences in our respective methods.



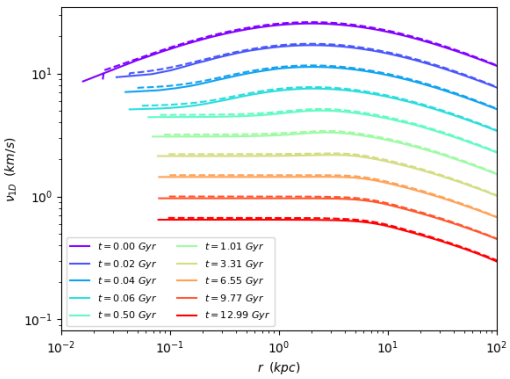
(a) Densities ($\sigma = 1\ cm^2/g$).



(b) v_{1D} ($\sigma = 1\ cm^2/g$).



(c) Densities ($\sigma = 10\ cm^2/g$).



(d) v_{1D} ($\sigma = 10\ cm^2/g$).

Figure 17: Comparison of `SPHerical` (solid) and Ref. [3] (dashed) results for ρ_i and $\nu_{1\mathrm{D}}$ at a variety of snapshot times. The ρ_i snapshots are offset by a factor of 10 and the $\nu_{1\mathrm{D}}$ snapshots by a factor of 1.5. `SPHerical` inputs were the same as those used in Figure 16.

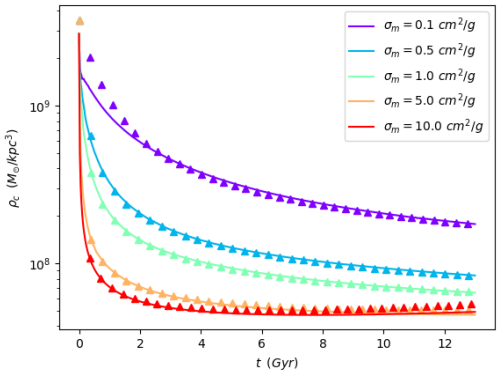


Figure 18: Comparison of `SPHerical` (solid) and Ref. [3] (triangle) ρ_c over time for a variety of cross sections. `SPHerical` inputs were the same as those used in Figure 16.

6 CONCLUSION

In this thesis, we discussed the need for a DM simulation package that can explore models beyond the standard CDM model. We then described how our new Python package `SPHerical` works and how it can be used to simulate the gravothermal evolution of isolated spherical SIDM halos. Lastly, we discussed how to choose input parameters for the package and presented results and comparisons with other simulations.

Our hope is that `SPHerical` will be used as a tool to test and study DM models that have not received as much attention as CDM. We are already working on including functionality in the package to explore particle physics models such as

- **Self-annihilating DM**: Just as the Standard Model contains particles and anti-particles, it is possible that a dark Standard Model might exists as well. If this were to be the case, DM particles would possess the ability to annihilate with one another. This process can be written as

  $$\chi\bar{\chi} \to \phi\phi$$

  where χ denotes a DM particle, ϕ denotes a dark boson (or force carrier particle), and "bar" denotes an antiparticle. If a DM halo consists of both particle and antiparticles, annihilation can serve as a mechanism by which the halo loses mass, thereby changing its structure. The densest regions of the halo would get depleted the greatest over time, as the annihilation cross section scales with density. See Ref. [72] for further reading.

- **Dissapative DM**: If DM particles and anti-particles can annihilate into dark bosons, then it may well be the case that they can also experience dissipations, losing energy via the emission of said bosons. For example, if there is such a thing as a dark electromagnetic force, then DM particles can radiate away dark photons. The dissipation process can be expressed as

  $$\chi\chi \to \chi\chi\phi \tag{255}$$

  describing a pair of DM particles scattering and losing energy via the emission of a dark boson. Since finite self-gravitating systems like DM halos have a negative heat capacity, losing energy via dissipation could actually serve to gravothermally collapse a halo, given a sufficiently large energy loss rate. Ref. [63] provides more details.

- **Multi-component DM**: DM might be self-interacting, self-annihilating, and dissipative. But what if DM consists of various types of particles, which display none, some, or all of the above phenomenon? If this were the case, there could be DM halos where a fraction f of the halo is comprised of, for example, dissipative DM, while the remaining $(1-f)$ is self-annihilating. There might also be instances of halos made of a portion of DM that self-interacts more strongly than the remainder, which can lead to gravothermal collapse and the formation of seed black holes in halo centers (see Ref. [73]).

REFERENCES

[1] Joseph Monaghan. "Smoothed Particle Hydrodynamics". In: *Reports on Progress in Physics* 68 (July 2005), p. 1703. DOI: 10.1088/0034-4885/68/8/R01.

[2] Daniel J Price. "Smoothed particle hydrodynamics and magnetohydrodynamics". In: *Journal of Computational Physics* 231.3 (2012), pp. 759–794.

[3] Hiroya Nishikawa, Kimberly K. Boddy, and Manoj Kaplinghat. "Accelerated core collapse in tidally stripped self-interacting dark matter halos". In: *Physical Review D* 101.6 (2020). ISSN: 2470-0029. DOI: 10.1103/physrevd.101.063009. URL: http://dx.doi.org/10.1103/PhysRevD.101.063009.

[4] A. Arbey and F. Mahmoudi. "Dark matter and the early Universe: A review". In: *Progress in Particle and Nuclear Physics* 119 (2021), p. 103865. ISSN: 0146-6410. DOI: 10.1016/j.ppnp.2021.103865. URL: http://dx.doi.org/10.1016/j.ppnp.2021.103865.

[5] Katherine Garrett and Gintaras Duda. "Dark Matter: A Primer". In: *Advances in Astronomy* 2011 (2011), 1–22. ISSN: 1687-7977. DOI: 10.1155/2011/968283. URL: http://dx.doi.org/10.1155/2011/968283.

[6] Stefano Profumo, Leonardo Giani, and Oliver F. Piattella. "An Introduction to Particle Dark Matter". In: *Universe* 5.10 (2019), p. 213. DOI: 10.3390/universe5100213. arXiv: 1910.05610 [hep-ph].

[7] Gianfranco Bertone, Dan Hooper, and Joseph Silk. "Particle dark matter: evidence, candidates and constraints". In: *Physics Reports* 405.5-6 (2005), 279–390. ISSN: 0370-1573. DOI: 10.1016/j.physrep.2004.08.031. URL: http://dx.doi.org/10.1016/j.physrep.2004.08.031.

[8] Jaan Einasto. "Dark Matter". In: *Brazilian Journal of Physics* 43.5-6 (2013), 369–374. ISSN: 1678-4448. DOI: 10.1007/s13538-013-0147-9. URL: http://dx.doi.org/10.1007/s13538-013-0147-9.

[9] Stephen M Kent. "Dark matter in spiral galaxies. I-Galaxies with optical rotation curves". In: *The Astronomical Journal* 91 (1986), pp. 1301–1327.

[10] Nick Kaiser and Gordon Squires. "Mapping the dark matter with weak gravitational lensing". In: *The Astrophysical Journal* 404 (1993), pp. 441–450.

[11] Scott Dodelson, Evalyn Gates, and Albert Stebbins. "Cold + hot dark matter and the cosmic microwave background". In: *Astrophys. J.* 467 (1996), pp. 10–18. DOI: 10.1086/177581. arXiv: astro-ph/9509147.

[12] Karsten Jedamzik and Maxim Pospelov. "Big Bang Nucleosynthesis and Particle Dark Matter". In: *New J. Phys.* 11 (2009), p. 105028. DOI: 10.1088/1367-2630/11/10/105028. arXiv: 0906.2087 [hep-ph].

[13] Hitoshi Murayama. "Physics Beyond the Standard Model and Dark Matter". In: *Les Houches Summer School - Session 86: Particle Physics and Cosmology: The Fabric of Spacetime*. Apr. 2007. arXiv: 0704.2276 [hep-ph].

[14] Andrew R Liddle and David H Lyth. "The Cold dark matter density perturbation". In: *Physics Reports* 231.1-2 (1993), pp. 1–105.

[15] Michael S. Turner. "The Case for ΛCDM". In: Mar. 1997. arXiv: astro-ph/9703161.

[16] G Efstathiou, Wo J Sutherland, and SJ Maddox. "The cosmological constant and cold dark matter". In: *Nature* 348.6303 (1990), pp. 705–707.

[17] Volker Springel et al. "Simulations of the formation, evolution and clustering of galaxies and quasars". In: *Nature* 435.7042 (2005), 629–636. ISSN: 1476-4687. DOI: 10.1038/nature03597. URL: http://dx.doi.org/10.1038/nature03597.

[18] Julio F Navarro. "The structure of cold dark matter halos". In: *Symposium-international astronomical union*. Vol. 171. Cambridge University Press. 1996, pp. 255–258.

[19] Ben Moore et al. "Resolving the structure of cold dark matter halos". In: *The Astrophysical Journal Letters* 499.1 (1998), p. L5.

[20] Liang Gao et al. "The Phoenix Project: the dark side of rich Galaxy clusters". In: *Monthly Notices of the Royal Astronomical Society* 425.3 (2012), pp. 2169–2186.

[21] Y. P. Jing. "The Density Profile of Equilibrium and Nonequilibrium Dark Matter Halos". In: *The Astrophysical Journal* 535.1 (2000), 30–36. ISSN: 1538-4357. DOI: 10.1086/308809. URL: http://dx.doi.org/10.1086/308809.

[22] Sean Tulin and Hai-Bo Yu. "Dark matter self-interactions and small scale structure". In: *Physics Reports* 730 (2018), 1–57. ISSN: 0370-1573. DOI: 10.1016/j.physrep.2017.11.004. URL: http://dx.doi.org/10.1016/j.physrep.2017.11.004.

[23] B. Moore. "Evidence against dissipationless dark matter from observations of galaxy haloes". In: *Nature* 370 (1994), p. 629. DOI: 10.1038/370629a0.

[24] A. Burkert. "The Structure of dark matter halos in dwarf galaxies". In: *Astrophys. J. Lett.* 447 (1995), p. L25. DOI: 10.1086/309560. arXiv: astro-ph/9504041.

[25] Matthew G Walker and Jorge Penarrubia. "A method for measuring (slopes of) the mass profiles of dwarf spheroidal galaxies". In: *The Astrophysical Journal* 742.1 (2011), p. 20.

[26] Se-Heon Oh et al. "High-resolution mass models of dwarf galaxies from LITTLE THINGS". In: *The Astronomical Journal* 149.6 (2015), p. 180.

[27] Ricardo A. Flores and Joel R. Primack. "Observational and theoretical constraints on singular dark matter halos". In: *The Astrophysical Journal* 427 (1994), p. L1. ISSN: 1538-4357. DOI: 10.1086/187350. URL: http://dx.doi.org/10.1086/187350.

[28] Manoj Kaplinghat, Sean Tulin, and Hai-Bo Yu. "Dark matter halos as particle colliders: unified solution to small-scale structure puzzles from dwarfs to clusters". In: *Physical Review Letters* 116.4 (2016), p. 041302.

[29] Tao Ren et al. "Reconciling the Diversity and Uniformity of Galactic Rotation Curves with Self-Interacting Dark Matter". In: *Physical Review X* 9.3 (2019). ISSN: 2160-3308. DOI: 10.1103/physrevx.9.031020. URL: http://dx.doi.org/10.1103/PhysRevX.9.031020.

[30] Michael Boylan-Kolchin, James S. Bullock, and Manoj Kaplinghat. "Too big to fail? The puzzling darkness of massive Milky Way subhaloes". In: *Monthly Notices of the Royal Astronomical Society: Letters* 415.1 (July 2011), pp. L40–L44. ISSN: 1745-3925. DOI: 10.1111/j.1745-3933.2011.01074.x. eprint: https://academic.oup.com/mnrasl/article-pdf/415/1/L40/2881682/415-1-L40.pdf. URL: https://doi.org/10.1111/j.1745-3933.2011.01074.x.

[31] Ben Moore et al. "Dark Matter Substructure within Galactic Halos". In: *The Astrophysical Journal* 524.1 (1999), L19–L22. ISSN: 0004-637X. DOI: 10.1086/312287. URL: http://dx.doi.org/10.1086/312287.

[32] Anatoly Klypin et al. "Where are the missing galactic satellites?" In: *The Astrophysical Journal* 522.1 (1999), p. 82.

[33] David H. Weinberg et al. "Cold dark matter: Controversies on small scales". In: *Proceedings of the National Academy of Sciences* 112.40 (2015), 12249–12255. ISSN: 1091-6490. DOI: 10.1073/pnas.1308716112. URL: http://dx.doi.org/10.1073/pnas.1308716112.

[34] James S. Bullock and Michael Boylan-Kolchin. "Small-Scale Challenges to the CDM Paradigm". In: *Annual Review of Astronomy and Astrophysics* 55.1 (2017), 343–387. ISSN: 1545-4282. DOI: 10.1146/annurev-astro-091916-055313. URL: http://dx.doi.org/10.1146/annurev-astro-091916-055313.

[35] James S. Bullock. "Notes on the Missing Satellites Problem". In: (Sept. 2010). arXiv: 1009.4505 [astro-ph.CO].

[36] Michael Boylan-Kolchin, James S. Bullock, and Manoj Kaplinghat. "The Milky Way's bright satellites as an apparent failure of CDM". In: *Monthly Notices of the Royal Astronomical Society* 422.2 (2012), 1203–1218. ISSN: 0035-8711. DOI: 10.1111/j.1365-2966.2012.20695.x. URL: http://dx.doi.org/10.1111/j.1365-2966.2012.20695.x.

[37] David N Spergel and Paul J Steinhardt. "Observational evidence for self-interacting cold dark matter". In: *Physical review letters* 84.17 (2000), p. 3760.

[38] Oliver D Elbert et al. "Core formation in dwarf haloes with self-interacting dark matter: no fine-tuning necessary". In: *Monthly Notices of the Royal Astronomical Society* 453.1 (2015), pp. 29–37.

[39] Jun Koda and Paul R Shapiro. "Gravothermal collapse of isolated self-interacting dark matter haloes: N-body simulation versus the fluid model". In: *Monthly Notices of the Royal Astronomical Society* 415.2 (2011), pp. 1125–1137.

[40] David N. Spergel and Paul J. Steinhardt. "Observational Evidence for Self-Interacting Cold Dark Matter". In: *Physical Review Letters* 84.17 (2000), 3760–3763. ISSN: 1079-7114. DOI: 10.1103/physrevlett.84.3760. URL: http://dx.doi.org/10.1103/PhysRevLett.84.3760.

[41] Shmuel Balberg, Stuart L. Shapiro, and Shogo Inagaki. "Self-Interacting Dark Matter Halos and the Gravothermal Catastrophe". In: *The Astrophysical Journal* 568.2 (2002), 475–487. ISSN: 1538-4357. DOI: 10.1086/339038. URL: http://dx.doi.org/10.1086/339038.

[42] Mark Vogelsberger et al. "Cosmological Simulations of Galaxy Formation". In: *Nature Rev. Phys.* 2.1 (2020), pp. 42–66. DOI: 10.1038/s42254-019-0127-2. arXiv: 1909.07976 [astro-ph.GA].

[43] Jazhiel Chacón, J. Alberto Vázquez, and Ruslan Gabbasov. "Dark matter with n-body numerical simulations". In: *Revista Mexicana de Física E* 17.2 Jul-Dec (2020), 241–254. ISSN: 1870-3542. DOI: 10.31349/revmexfise.17.241. URL: http://dx.doi.org/10.31349/RevMexFisE.17.241.

[44] Go Ogiya et al. "Studying the core-cusp problem in cold dark matter halos using N-body simulations on GPU clusters". In: *Journal of Physics: Conference Series*. Vol. 454. 1. IOP Publishing. 2013, p. 012014.

[45] Robert J. J. Grand et al. "The Auriga Project: the properties and formation mechanisms of disc galaxies across cosmic time". In: *Monthly Notices of the Royal Astronomical Society* (2017), stx071. ISSN: 1365-2966. DOI: 10.1093/mnras/stx071. URL: http://dx.doi.org/10.1093/mnras/stx071.

[46] Nima Arkani-Hamed et al. "A theory of dark matter". In: *Physical Review D* 79.1 (2009). ISSN: 1550-2368. DOI: 10.1103/physrevd.79.015014. URL: http://dx.doi.org/10.1103/PhysRevD.79.015014.

[47] Joop Schaye et al. "The EAGLE project: Simulating the evolution and assembly of galaxies and their environments". In: *Mon. Not. Roy. Astron. Soc.* 446 (2015), pp. 521–554. DOI: 10.1093/mnras/stu2058. arXiv: 1407.7040 [astro-ph.GA].

[48] Till Sawala et al. "The APOSTLE simulations: solutions to the Local Group's cosmic puzzles". In: *Monthly Notices of the Royal Astronomical Society* 457.2 (2016), 1931–1943. ISSN: 1365-2966. DOI: 10.1093/mnras/stw145. URL: http://dx.doi.org/10.1093/mnras/stw145.

[49] M. Vogelsberger et al. "Properties of galaxies reproduced by a hydrodynamic simulation". In: *Nature* 509.7499 (2014), 177–182. ISSN: 1476-4687. DOI: 10.1038/nature13316. URL: http://dx.doi.org/10.1038/nature13316.

[50] Yohan Dubois et al. "Dancing in the dark: galactic properties trace spin swings along the cosmic web". In: *Monthly Notices of the Royal Astronomical Society* 444.2 (2014), pp. 1453–1468.

[51] J. J. Monaghan. "Smoothed Particle Hydrodynamics". In: *Annual Review of Astronomy and Astrophysics* 30.1 (1992), pp. 543–574. DOI: 10.1146/annurev.aa.30.090192.002551. eprint: https://doi.org/10.1146/annurev.aa.30.090192.002551. URL: https://doi.org/10.1146/annurev.aa.30.090192.002551.

[52] Michael Boylan-Kolchin et al. "Resolving cosmic structure formation with the Millennium-II Simulation". In: *Monthly Notices of the Royal Astronomical Society* 398.3 (2009), 1150–1164. ISSN: 1365-2966. DOI: 10.1111/j.1365-2966.2009.15191.x. URL: http://dx.doi.org/10.1111/j.1365-2966.2009.15191.x.

[53] Philip F Hopkins et al. "FIRE-2 simulations: physics versus numerics in galaxy formation". In: *Monthly Notices of the Royal Astronomical Society* 480.1 (2018), 800–863. ISSN: 1365-2966. DOI: 10.1093/mnras/sty1690. URL: http://dx.doi.org/10.1093/mnras/sty1690.

[54] Jesus Prada et al. "Dark matter halo shapes in the Auriga simulations". In: *Monthly Notices of the Royal Astronomical Society* 490.4 (2019), 4877–4888. ISSN: 1365-2966. DOI: 10.1093/mnras/stz2873. URL: http://dx.doi.org/10.1093/mnras/stz2873.

[55] Jo Bovy et al. "The shape of the inner milky way halo from observations of the Pal 5 and GD–1 Stellar Streams". In: *The Astrophysical Journal* 833.1 (2016), p. 31.

[56] M Omang, Steinar Børve, and Jan Trulsen. "SPH in spherical and cylindrical coordinates". In: *Journal of Computational Physics* 213.1 (2006), pp. 391–412.

[57] J. J. Monaghan. "Smoothed Particle Hydrodynamics". In: *Annual Review of Astronomy and Astrophysics* 30.1 (1992), pp. 543–574. DOI: 10.1146/annurev.aa.30.090192.002551. eprint: https://doi.org/10.1146/annurev.aa.30.090192.002551. URL: https://doi.org/10.1146/annurev.aa.30.090192.002551.

[58] Sydeny Chapman and T. G. Cowling. *The mathematical theory of non-uniform gases. an account of the kinetic theory of viscosity, thermal conduction and diffusion in gases.* 1970.

[59] Walter Guido Vincenti and Charles H. Kruger. *Introduction to physical gas dynamics.* 1965.

[60] D. Lynden-Bell and P. P. Eggleton. "On the consequences of the gravothermal catastrophe". In: *Monthly Notices of the Royal Astronomical Society* 191.3 (July 1980), pp. 483–498. ISSN: 0035-8711. DOI: 10.1093/mnras/191.3.483. eprint: https://academic.oup.com/mnras/article-pdf/191/3/483/2913541/mnras191-0483.pdf. URL: https://doi.org/10.1093/mnras/191.3.483.

[61] S. Tulin. *Dark matter self-interactions and small scale structure.* Presented at Workshop on Perspectives on the ICTP's Extragalactic Frontier: from Astrophysics to Fundamental Physics, Miramare, Trieste, Italy, 2016.

[62] Shmuel Balberg, Stuart L. Shapiro, and Shogo Inagaki. "Self-Interacting Dark Matter Halos and the Gravothermal Catastrophe". In: *The Astrophysical Journal* 568.2 (2002), 475–487. ISSN: 1538-4357. DOI: 10.1086/339038. URL: http://dx.doi.org/10.1086/339038.

[63] Rouven Essig et al. "Constraining Dissipative Dark Matter Self-Interactions". In: *Physical Review Letters* 123.12 (2019). ISSN: 1079-7114. DOI: 10.1103/physrevlett.123.121102. URL: http://dx.doi.org/10.1103/PhysRevLett.123.121102.

[64] F. Reif. *Fundamentals of Statistical and Thermal Physics.* Waveland Press, Long Grove, IL, 2009.

[65] Martin Jubelgas, Volker Springel, and Klaus Dolag. "Thermal conduction in cosmological SPH simulations". In: *Monthly Notices of the Royal Astronomical Society* 351.2 (2004), 423–435. ISSN: 1365-2966. DOI: 10.1111/j.1365-2966.2004.07801.x. URL: http://dx.doi.org/10.1111/j.1365-2966.2004.07801.x.

[66] Leigh Brookshaw. "Solving the heat diffusion equation in SPH". In: *Memorie della Societa Astronomica Italiana* 65 (1994), p. 1033.

[67] Stephan Rosswog. "Astrophysical smooth particle hydrodynamics". In: *New Astronomy Reviews* 53.4-6 (2009), 78–104. ISSN: 1387-6473. DOI: 10.1016/j.newar.2009.08.007. URL: http://dx.doi.org/10.1016/j.newar.2009.08.007.

[68] Matthew R Bate, Ian A Bonnell, and Nigel M Price. "Modelling accretion in protobinary systems". In: *Monthly Notices of the Royal Astronomical Society* 277.2 (1995), pp. 362–376.

[69] J.J. Monaghan. "SPH without a Tensile Instability". In: *Journal of Computational Physics* 159.2 (2000), pp. 290–311. ISSN: 0021-9991. DOI: https://doi.org/10.1006/jcph.2000.6439. URL: https://www.sciencedirect.com/science/article/pii/S0021999100964398.

[70] Vishal Mehra et al. "Tensile instability and artificial stresses in impact problems in SPH". In: *Journal of Physics: Conference Series*. Vol. 377. 1. IOP Publishing. 2012, p. 012102.

[71] J. I. Read, T. Hayfield, and O. Agertz. "Resolving mixing in smoothed particle hydrodynamics". In: *Monthly Notices of the Royal Astronomical Society* 405.3 (June 2010), pp. 1513–1530. ISSN: 0035-8711. DOI: 10.1111/j.1365-2966.2010.16577.x. eprint: https://academic.oup.com/mnras/article-pdf/405/3/1513/2875672/mnras0405-1513.pdf. URL: https://doi.org/10.1111/j.1365-2966.2010.16577.x.

[72] Manoj Kaplinghat, Lloyd Knox, and Michael S. Turner. "Annihilating Cold Dark Matter". In: *Physical Review Letters* 85.16 (2000), 3335–3338. ISSN: 1079-7114. DOI: 10.1103/physrevlett.85.3335. URL: http://dx.doi.org/10.1103/PhysRevLett.85.3335.

[73] Jason Pollack, David N. Spergel, and Paul J. Steinhardt. "SUPERMASSIVE BLACK HOLES FROM ULTRA-STRONGLY SELF-INTERACTING DARK MATTER". In: *The Astrophysical Journal* 804.2 (2015), p. 131. ISSN: 1538-4357. DOI: 10.1088/0004-637x/804/2/131. URL: http://dx.doi.org/10.1088/0004-637X/804/2/131.