Halo and Sub-Halo Finding in Cosmological N-body Simulations

Mladen Ivkovic

Institute for Computational Science
University of Zurich
Switzerland

June 2017

A thesis presented for the degree of
Bachelor of Science in Physics

Supervisor: Prof. Dr. Romain Teyssier

Contents

Abstract

The implementation of a particle unbinding algorithm into the adaptive mesh refinement (AMR) code `RAMSES`, which uses the segmentation algorithm `PHEW` to identify halos and their substructure in simulations of collisionless dark matter is presented. Only particles initially assigned to subhalos are examined, particles of halos are assumed to be bound. The algorithm follows the hierarchy of substructures as found by `PHEW` and passes unbound particles on to the next higher level of substructure for examination. Furthermore, it accounts for neighbouring structures by demanding a particle to be exclusively bound to the subhalo that is being investigated. In order to increase the accuracy of the substructure's properties, they are determined iteratively by taking into account only bound particles.
The algorithm is fully parallel and uses the MPI library. Its effects, accuracy and resource usage are shown on two highly idealistic datasets and a cosmological simulation.

1. Introduction

1.1. Halo Finding

With growing processing power and the use of parallel computing tools, numerical codes designed for the simulation of cosmic structure are not only highly reliable and cost effective, but much larger and more elaborate simulations are becoming possible. In order to compare the results of cosmological simulations to observations of the Universe, the simulation data needs to be analysed. For simulations of collisionless dark matter in particular, an important tool for problems concerning cosmic structure and its formation is the identification of *halos*, i.e. gravitationally bound objects made of particles as well as their internal structure and bound objects nested within them, called *subhalos*. Codes that perform this task are called "*halo-finders*".
While the simplest way to define a halo would probably be as a "gravitationally bound object", such a definition is ambiguous. Whether a particle is gravitationally bound to an object depends on the mass of said object, so removing unbound particles might lead to even more unbound particles since it reduces the object's mass. This is particularly important for subhalos, which usually consist of much less particles than their hosts. Additionally, another question is how to treat the nested subhalos: Do their properties like mass and momentum contribute to the properties of the host, or are subhalos completely separate structures?
Yet another difficulty with subhalos is a definition of their edge. Usually, the edge of a halo is defined by some requirement of overdensity, but even that is ambiguous: the overdensity is the product of a parameter and a reference density. One might choose varying parameters as well as differing reference densities according to different theoretical models or applications. This makes the definition of the edge of subhalos even more difficult: How exactly should the edge of subhalos be defined, which are not in isolation, but located within the host's density field? How does one distinguish between substructure and statistical noise of a density field? There is no unique answer to these questions, as different problems and applications make use of different definitions (Knebe et al. 2013).

A further requirement for substructure-finding is the removal of energetically unbound particles, i.e. assigning a particle originally located within a substructure to the parent structure based on an energy criterion. This applies recursively to any level of substructure within substructure. Such a hierarchical clustering of matter is expected due to self gravity. It is customary to treat all particles assigned to a halo as bound to it, even though from a strict energetic perspective

they are not, thus particle unbinding may not be necessary for halos, but it is vital for subhalos. Subhalos are by definition located within a host halo and are therefore expected to be contaminated by the host's particles. Considering that substructure often contains far less particles than their hosts, blindly assigning particles to it without an unbinding procedure can influence its physical properties significantly.

1.2. Halo- and Subhalo-Finders

Over the last decades, a multitude of halo finding tools has been introduced. The Halo-Finder Comparison Project (Knebe et al. 2011) lists 29 different codes in the year 2010 and roughly divides them into two distinct groups of codes:

1. particle collector codes, where particles are linked together
2. density peak locator codes, that first find density peaks and then collect particles around those.

Particle collector codes most commonly use some form of the "*Friends-of-Frieds*" (FOF) method (Davis et al. 1985). Groups are identified by linking "friends" together, i.e. particles that are closer to each other then some specified linking length. A particle belongs to a group if it has friends within this group. This implicitly determines some minimal density for the halos found this way. The particles may be linked either in 3D space or in 6D phase space. The advantage of phase space finders like `ROCKSTAR` (Behroozi, Wechsler, and Wu 2012) is that they are able to identify halo centres even if they overlap with another object.

Density peak locator codes identify halos and subhalos around previously found peaks in the mass density field. One frequently used method to identify halos in such manner is the "*Spherical Overdensity*" (SO) method (Press and Schechter 1974). The basic idea is to find groups of particles by growing spherical shells around those peaks until the mean overdensity of the sphere falls below some threshold.
But there are also other approaches: `PHEW` (Bleuler et al. 2015) for example, which will be discussed in greater detail in section 2.1.2, assigns cells (not particles) to density peaks following the steepest density gradient. Unlike the SO method, this allows to identify halos without the assumption of spherical symmetry.

Various techniques to identify unbound particles within the found structures are used as well. Because removing particles from a structure changes said structures properties, often iterative approaches are used. `AHF` (Knollmann and Knebe 2009), `ASOHF` (Planelles, S. and Quilis, V. 2010) and `SUBFIND` (Springel et al. 2001) for example remove all unbound particles from a halo, recompute the potential and repeat the procedure until no more particles are removed. Unbound particles are then passed from subhalos on to host halos (or host subhalos) for examination. `SKID` (Stadel 2001) however only removes the particle with the highest energy per iteration.

1.3. On-the-fly Analysis

While bigger simulations can yield more precise data, they also produce large amounts of data which needs to be stored and post-processed effectively. This creates a variety of issues. On one hand there is a possibility that not all produced simulation data can be stored because it is simply too large. Another issue is that most modern astrophysical simulations are executed on large supercomputers which offer large distributed memory. Post-processing the data they

produce may also require just as much memory, so that the analysis will also have to be executed on the distributed memory infrastructures. The reading and writing of a vast amount of data to a permanent storage remains a considerable bottleneck, particularly so if the data need to be read and written multiple times.
One way to reduce the computational cost is to include analysis tools like halo-finders in the simulations and run them "*on-the-fly*", i.e. run them during the simulation. This allows to store only the interesting parts and regions instead of the full raw data. A caveat of this method is that the halo-finder should be "fast enough" with respect to the execution time of the simulation. An advantage, on the other hand, is that it offers the option to implement the results of the halo-finders into the simulation and further analysis.

The segmentation algorithm `PHEW` implemented in `RAMSES` (Teyssier, R. 2002) is precisely such an analytical tool which allows an on the fly analysis of the simulation domain. In this thesis a newly developed particle unbinding code designed to work on-the-fly within the framework of `PHEW` and `RAMSES` is presented.

This thesis is structured as follows. In the next chapter `RAMSES`, `PHEW` and their parallel implementation with emphasis placed on aspects of the codes that are closely related and necessary for the unbinding code are described shortly. Then physical aspects of particle unbinding are discussed. For the sake of clarity, there is a small glossary in appendix A. In chapter 3 a detailed description of the unbinding code is given. Chapter 4 introduces the datasets on which the unbinding code will be run on. The results of different available unbinding methods, the accuracy in dependence of parameters and the resource usage of the code are given in chapter 5, followed by a conclusion in chapter 6.

2. Theory

2.1. Overlying Framework: RAMSES and PHEW

2.1.1. RAMSES

`RAMSES` is a N-body and hydrodynamical adaptive mesh refinement (AMR) code which uses the "Fully Threaded Tree" data structure of Khokhlov (Khokhlov 1998). It simulates the time evolution of particles. The domain is covered by a Cartesian grid, called the mesh. Simulations require numerical integration, whose accuracy increases as the size of a grid cell decreases, but smaller grid cells naturally need more resources to cover a domain of the same size. AMR is a method that "*tries to attain a fixed accuracy for a minimum cost*" (Berger and Colella 1989) by "refining" the mesh only where and when necessary. "Refining" in this case means that cells of smaller and smaller sizes are introduced until the desired accuracy is achieved. Grid cells that contain lots of particles will be refined many times, while grid cells that contain no particles won't be refined at all. The unrefined grid is called the "*coarse grid*" and the number of times a cell of the coarse grid is refined is called the "level of refinement". Cells which are not refined (any further) are called "*leaf cells*". They always have the highest level of refinement at their position.

The basic elements of the data structure in `RAMSES` are groups of 2^{dim} sibling cells called "*octs*". Each oct belongs to a given level of refinement. All octs of a given refinement level are sorted in a doubly linked list, so each oct points to the previous and the next oct in the linked list of the particular level. The fully threaded tree structure also demands that each oct points to the

parent cell (the cell on the less refined level) as well as the $2 \cdot \mathrm{dim}$ neighbouring parent cells and the 2^{dim} child octs on the next refinement level, assuming they exist.
Particles belonging to octs are organised in linked lists as well. A particle belongs to a particular oct if its position fits exactly into the oct boundaries, and all particles that belong to the same oct are linked together by a linked list.

A system that contains many particles is called a "N-body system". A collisionless N-body system is described by the following equations for the particles with position $\vec{x}_p$ and velocity $\vec{v}_p$:

$$\frac{\mathrm{d}\vec{x}_p}{\mathrm{d}t} = \vec{v}_p \qquad \frac{\mathrm{d}\vec{v}_p}{\mathrm{d}t} = -\nabla_x \phi \tag{1}$$

with

$$\Delta_x \phi = 4\pi G \rho \tag{2}$$

Where ϕ is the potential, G the gravitational constant and ρ the mass density field. To compute the spatial movement of the particles, first ρ is computed using a "*Cloud-In-Cell*" (CIC) interpolation scheme. The CIC scheme considers all particles to be cubes ("clouds") of one cell size and of uniform density. The mass of a particle is deposited in cells based on what fraction of its "cloud" overlaps with the cell, thus determining the density field. Once the density field is known, the Poisson equation (2) can be solved numerically and the potential ϕ is computed. Now the acceleration for each cell can be obtained, from which the particle acceleration $\frac{\mathrm{d}\vec{v}_p}{\mathrm{d}t}$ is computed using the inverse CIC interpolation. The acceleration is then integrated over time to compute the particle velocity $\vec{v}_p$ and position $\vec{x}_p$ using a second-order midpoint scheme.

2.1.2. PHEW

`PHEW` is a structure finding algorithm implemented into `RAMSES`. It is a "watershed-based" algorithm: "*These algorithms assign particles or cells to density peaks by following the steepest gradient, resulting in the so called 'watershed segmentation' [...] of the negative density field.*" (Bleuler et al. 2015). `PHEW` groups cells together by separating the mass density field along minima, thus dividing the density field into patches. The density field is obtained with the deposition of the particle's mass on the mesh through CIC interpolation. The algorithm can be divided in four main steps: segmentation, connectivity establishment, noise removal and substructure merging.

In the first step, namely the "watershed" segmentation, the cells of interest are identified. These are all leaf cells which have a density above an user-defined density threshold and are called "*test cells*". If a cell doesn't have a denser neighbour, it is marked as a local density peak and assigned a peak label. The peak label of a cell defines what peak patch this particular cell belongs to.
Then each cell copies the peak label from its densest neighbour. This way, each cell is assigned to the peak "closest" to it by following the path of rising density. All test cells assigned to a particular peak form the aforementioned peak patch. The separating surface between peak patches will be cell borders which contain local density minima, corresponding to the watershed analogy.

In the next step, the connections between the peak patches need to be established. All test cells are examined for neighbours that belong to another peak. If a neighbouring cell with a different peak label is found, the density of the common surface between those two neighbours is defined as the average of these two cell densities. The maximal surface density between two particular peak patches is considered as the "*saddle*" between these two. Naturally, a peak patch can have

multiple neighbouring peak patches. Out of all the saddles of all the neighbouring peak patches, the one with the highest density is called the "*key saddle*" and the neighbour it connects to is referred to as the "*key neighbour*".

The third main step, noise removal, requires a clear definition of what is to be considered noise. In `PHEW`, each peak patch is assigned a value representing the contrast to the background called "*relevance*". A peak patch's relevance is defined as the ratio of the peak's density to its key saddle. A peak patch is considered noise if the its relevance is lower than a user-defined relevance threshold. An irrelevant peak patch is then merged to its key neighbour. If an irrelevant peak patch doesn't have a key neighbour because it is isolated, it is discarded.

Expressed explicitly, merging a peak patch i into a peak j means that all cells of i inherit the peak label of j. Both peaks' saddles are merged as well: The peak patch j inherits the saddles that connected i to other patches. Therefore, after two peaks are merged, new key saddles and the relevance need to be determined for the peak patches that have undergone merging. This concludes the noise removal, since all irrelevant patches are either merged into relevant peaks or discarded.
As mentioned before, `PHEW` can work in parallel on multiple processors. The parallel implementation will be discussed in more detail in section 2.1.3. In order to produce unique results when executed on multiple processors, the merging of peak patches needs to obey the following three rules:

1. Whether a peak should be merged into its key neighbour or not is determined by some criterion.
2. A peak patch is only merged into its key neighbour.
3. The peak of the key neighbour must have a higher density.

In the case of noise removal, the criterion is to satisfy the relevance condition. The criterion is different for the substructure merging step.
Once the noise removal step is completed, the remaining structure consists only of peak patches which satisfy the relevance condition and is referred to as "*level 0 clumps*". These clumps represent the structure on the lowest scale. A large halo for example, which can very roughly be described as "a large clump" in a first approximation, would be decomposed into many small clumps. In order to identify such a halo as a single object, one more step is necessary.

The identified level 0 clumps can be merged further into composite clumps. The procedure used is exactly the same as in the previous step, only the merging criterion is different: The new criterion is a threshold for the key saddle density. All peak patches whose key saddle density is *higher* than the user-defined threshold are merged into their key neighbour. The saddle threshold defines which clumps should be considered as separate structures and which should be merged and considered as composite structures.
After the first round of merging, the lowest level of structure has been merged into first composite clumps. If clump a is merged into clump b, then b is called the "*parent*" of a. Because of the merging, again the lists of saddle points change for the "surviving" clumps, and the list as well as the key saddles need to be updated. The peak labels of the cells however are not updated in this merging procedure: They are necessary to preserve the information on substructure. Only the clump properties like volume, mass, position and neighbours are merged.
The merging of clumps creates connections between peak patches where there were none before in the sense that the surviving peak patches can establish a connection to other surviving patches because clumps between them were merged into them. See for example in figure 1: The red and the blue clump establish a connection after all the smaller clumps between them have been merged into them in the span of two merging loops. Each merging loop then represents a

different level of substructure: In each loop, greater structures are being merged into each other than there were in the loop before. Starting from the lowest level, which are the peak patches identified after noise removal, clumps are merged into each other until none satisfy the merging criterion any more.

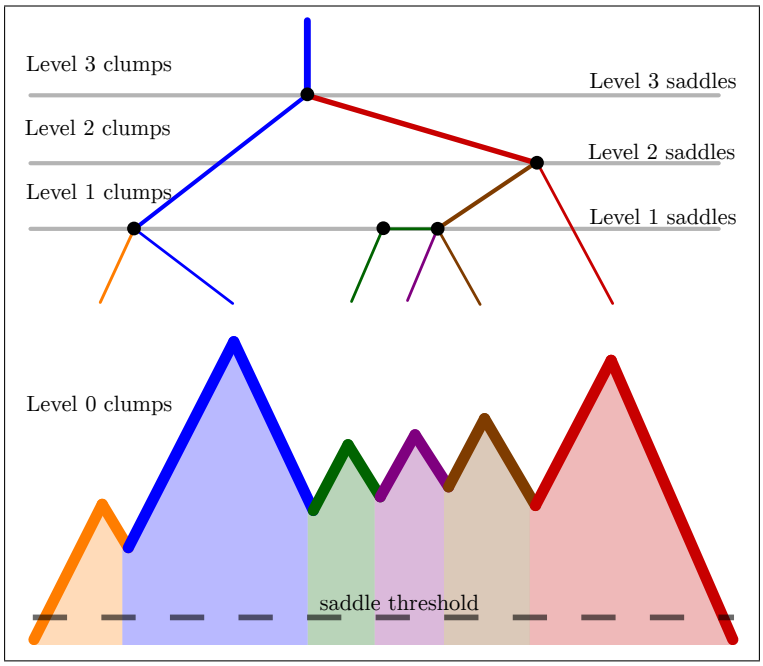

Figure 1: Hierarchy of clumps as created by PHEW's clump merging algorithm. Level n saddle points are used for merging during the $n-$th round of merging. A peak patch is always merged into its key neighbour and it is only merged into a peak patch with a higher density. In this figure, the merging criterion is that the key saddle is higher than the saddle threshold.
Image adapted from Bleuler et al. 2015

When the merging is finished, the clumps have been merged into the largest possible structures that satisfy the relevance and saddle density conditions. The remaining structures are large clumps and they are considered as halos, provided that the clumps are massive enough. A halo must have a minimal mass that can be set by the user.
The label of halos will be the label of clumps which are never merged into an other, but only have other clumps merged into them. Such clumps are called "*halo-namegivers*".

In summary, PHEW identifies halos by the following three criteria:

- Halos are composed of cells with densities above a density threshold.
- A halo must have a minimal mass that can be set by the user.
- Within a halo, the subhalos are peak patches that are relevant, i.e. aren't considered noise. Their edge is determined by local density minima surfaces.
- If the interface between two relevant peak patches is less dense than a threshold (saddle threshold), the two peak patches are considered to be two independent halos.

2.1.3. Parallel implementation

RAMSES is a parallel code which makes use of the MPI library. The use of MPI (message passing interface) allows a process on a distributed memory architecture to be executed in parallel by multiple tasks and enables various types of communications between them.

The fundamental parallelisation strategy used in `RAMSES` is domain decomposition, where a part of the total spatial computational domain is assigned to each processing unit, or "MPI task". It makes use of the fact that most calculations on grids do not require the knowledge of the entire computational domain, but only the cells in their vicinity The basic idea of the domain decomposition is illustrated in figure 2, where a 2D-grid is split between two processors. The partial domains do not overlap and a thin layer of cells called the "*virtual boundary*" is introduced where the domain was cut. Necessary information is then communicated across MPI tasks from the other tasks' "real domain" into the virtual boundary, so that the virtual boundary copies what happens in the domains of other tasks and allowing the execution of the code as if the domain wasn't split. Two tasks working on the same problem allows a much faster execution time, but it also enables the solution of a much bigger problem that only one task couldn't do on its own, e.g. because of memory restrictions.

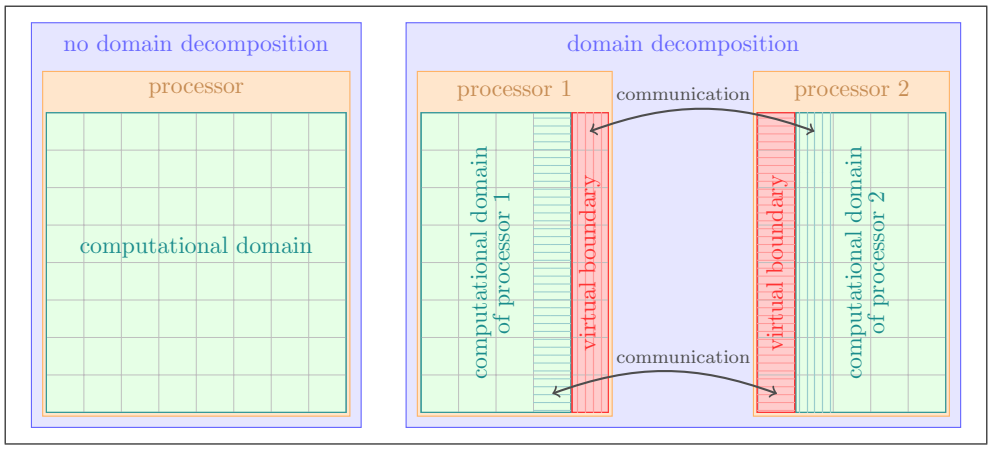

Figure 2: The basic idea of domain decomposition. Here a 2D-grid is split between two processors (right) instead of only one (left). The partial domains do not overlap and where the domain was cut, a "virtual boundary" is introduced. Necessary information is then communicated between processors from the other tasks' "real domain" into the virtual boundary, so that the virtual boundary copies what happens in the domains of other tasks and allows the execution of the code as if the domain wasn't split.

`PHEW` uses the virtual mesh boundary as well, since every cell on each domain must have the information of all its neighbours. Initially, all peaks are counted and their number is communicated throughout all MPI tasks. This gives all MPI tasks the knowledge of the total number of peaks of the entire computational domain, allowing the introduction of a peak label that is globally unique for each peak.

Similarly to the virtual mesh boundary, a virtual peak boundary is necessary. For the peak patch merging step, each peak patch on each task's domain needs the information of the peak patches that surround it. If, for example, a peak patch is split in two by the domain boundaries between two tasks, both tasks need to know all the peak patch's neighbours on the other task's domain.
Unlike the mesh boundary however, the virtual peak boundary is not a fixed region in space. Because of the merging, peaks gain neighbours they hadn't had before. This requires that new peaks are introduced to the virtual peak boundary during the merging procedure. Once introduced, all other peak properties, e.g. its relevance and saddle points, can be transferred by means of MPI communication.

The virtual peak boundary requires two types of communications. One type is the collection (sum, minimum or maximum) of a value for a peak from all tasks which have that particular peak patch in their virtual boundary to the owner of the peak. The "*owner*" of the peak is the task where the density peak is in the "real" domain, as opposed to the virtual mesh boundary.

Imagine for example the calculation of a peak patch's total volume on multiple tasks: Each task would compute the volume of that peak patch on its own domain and then all these partial results would be sent to the peak's owner and summed up.
In that scenario only the peak's owner has the total volume of the patch, but the ones that have it in their virtual boundary still only have their partial values. This brings us to the second required type of communication: A scatter of data from the owner of the peak to all tasks with that particular peak in their virtual boundary. In our example, the owner of the peak would send the computed total volume of the peak patch to all the tasks which require that information. In order to perform these communications, a communication structure (called the "peak communicator") must be built first. The purpose of the peak communicator is to establish how many peaks of every task are owned by any other task, or in other words: "what needs to be sent (and received from) where". Once that is known, the communications between the processes and thus the parallel merging can be performed.

The following summary of results and tools available after the execution of the segmentation algorithm concludes this brief description of `PHEW` and `RAMSES`:

- A list of peak patches, composed of overdense leaf cells, representing clumps of particles
- A hierarchy of these clumps, established during the merging of the peak patches
- A virtual boundary for cells and another virtual boundary for peak patches along with as a set of communication tools which enable the transfer of peak patch information across MPI tasks.

2.2. Unbinding Particles

2.2.1. Constraints and Limitations

Due to complexity and size, solutions of physical problems often require simplifications and assumptions to be made. In this section, the assumptions necessary for the unbinding algorithm are explained shortly.

Firstly, the situation under consideration is assumed to be *time-independent*. This allows an easy description of the system by introducing local energy conservation on one hand, but on the other hand, the information necessary for a time-dependent description is not readily obtainable. `RAMSES` computes the evolution of the particles time step by time step, thus creating a snapshot of the particles' position and velocity at each time step. `PHEW` can then be executed on the results of the time advancement, meaning that the segmentation is only performed on snapshots of the simulation. In order to obtain the information necessary for a time-dependent description, the entire simulation from beginning to end must be known, which contradicts the requirement for on-the-fly analysis. Furthermore, due to the nature of `PHEW`, the found peak patches can't be tracked trough time: Only one more or one fewer local density maximum would shift all following peak labels.

The purpose of a particle unbinding algorithm is to determine for each particle of a (sub)structure if the particle is bound to that particular structure. The focus on the substructure itself, as opposed to a global perspective, leads to the second used simplification: The structure is considered to be *isolated*. Any effects on the particles that arise from outside the structure that is being investigated, e.g. tidal effects, are neglected. (Some consequences of neighbouring structures will be accounted for, which is will be discussed in section 2.2.3). This simplification is also motivated by the enormous computational expense that would arise from taking the entirety of structures in the computational domain into account for each substructure anew.

Lastly, a third major simplification is used. As will be introduced more properly in the following section, whether particles are bound to a clump depends on the gravitational potential of the clump itself. In classical mechanics, the gravitational potential $\phi(\vec{r})$ is determined by the poisson equation 2. It represents the amount of work needed to move the particle from infinity (where as a point of reference the potential is usually chosen to vanish) to the place it is.
In theory, equation 2 can be solved, because if all the particle positions are known, then the density field ρ is known as well. Sadly the solution for each particle depends on each other particle, so calculating it makes it a computationally expensive code of order $\mathcal{O}(N^2)$. This is not acceptable for a code required to work on-the-fly. Instead, an analytical solution which is valid for *spherically symmetric* systems is used. It is a bit unfortunate to introduce the assumption of spherical symmetry, because `PHEW` had no need for it and was able to identify clumps of any shape, an information which is ignored at this point. Nonetheless, due to the lack of a better solution at this time, it was a necessity.

To summarise, each substructure clump as found by `PHEW` is considered to be an isolated, time independent spherical system of collisionless dark matter particles.

2.2.2. Bound Particles

In general, one can assign every particle a total energy E. A particle is considered to be "*bound*" if

$$E < 0 \tag{3}$$

or "*unbound*" if it doesn't satisfy this condition.

Consider now an isolated clump that is made up of N particles, where the position $\vec{x}_i$, the velocity $\vec{u}_i$ and the mass m_i of every particle i is known. Even though the system is assumed to be time independent, the structures within the system do not need to be stationary. On the contrary: The clumps are expected to be in motion with respect to the computational grid. It is often useful to consider the problem in another frame of reference, namely the "centre of mass" frame. The centre of mass of a clump of N particles is defined as:

$$\vec{x}_{CoM} = \frac{\sum_{i=1}^{N} m_i \cdot \vec{x}_i}{\sum_{i=1}^{N} m_i} \tag{4}$$

The centre of mass frame is then defined as the frame of reference where the velocity of the centre of mass (also called the "*bulk velocity*") vanishes:

$$\vec{v}_{CoM} = \frac{\mathrm{d}}{\mathrm{d}t}\vec{x}_{CoM} = \frac{\sum_{i=1}^{N} m_i \cdot \vec{u}_i}{\sum_{i=1}^{N} m_i} \equiv 0 \qquad \text{in centre of mass frame} \tag{5}$$

and the centre of mass is at the origin:

$$\vec{x}_{CoM} \equiv 0 \tag{6}$$

The particle positions and velocities in the centre of mass frame are then

$$\vec{x_i}\,' = \vec{x}_i - \vec{x}_{CoM} \equiv r_i \tag{7}$$

$$\vec{u}_i\,' = \vec{u}_i - \vec{v}_{CoM} \equiv \vec{v}_i \qquad \text{with} \qquad v_i \equiv ||\vec{v}_i|| = \sqrt{v_{x,i}^2 + v_{y,i}^2 + v_{z,i}^2} \tag{8}$$

The assumption of time-independence and isolation allow to describe the clump by means of classical mechanics. In such a system the energy E_i of a particle i in the centre of mass frame of the clump it belongs to can be expressed as

$$E_i = T_i + V_i = \frac{1}{2} m_i \cdot v_i^2 + m_i \phi(\vec{r}_i) \tag{9}$$

and is conserved:

$$E_i = const \tag{10}$$

Where $T_i = \frac{1}{2} m_p v_i^2$ is the *kinetic energy* and $V_i = m_i \cdot \phi(\vec{r}_i)$ is called the *potential energy* of the particle. $\phi(\vec{r}_i)$ is called the *potential*.

Plugging equation (9) into the condition for a particle to be bound (3), we get

$$\frac{1}{2} m_i \cdot v_i^2 + m_i \phi(\vec{r}_i) < 0 \tag{11}$$

Or equivalently:

$$v_i < \sqrt{-2 \cdot \phi(\vec{r}_i)} \tag{12}$$

These equations allow for an evident physical interpretation of the criterion for a particle to be bound: It is bound if its kinetic energy is lower than the absolute value of the (gravitational) potential that it experiences. A bound particle's movement will be constrained by the potential: The potential will always pull the particle towards the centre of mass of the clump. It can not escape arbitrarily, but will orbit the clump on elliptic trajectories.

$\vec{r}_i$ and $\vec{v}_i$ are known for each particle. What remains to be found is an expression for the potential ϕ. Since only isolated collisionless dark matter particles are considered, the potential that influences the particles is the gravitational potential of the particles themselves. In a spherically symmetric system, the potential at each point is determined by the absolute (radial) distance as opposed to its exact location. Thus the potential can be expressed in the clump's centre of mass frame as $\phi(\vec{r}_i) = \phi(d_i)$ with

$$d_i = \sqrt{(r_{i,x} - x_{CoM,x})^2 + (r_{i,y} - x_{CoM,y})^2 + (r_{i,z} - x_{CoM,z})^2} \tag{13}$$

The spherically symmetric Poisson equation (eq. 2) can be solved for ϕ:

$$\phi(r_i) = -G \int_{r_i}^{r_{max}} \frac{M(< \tilde{r})}{\tilde{r}^2} \mathrm{d}\tilde{r} - G \frac{M_{tot}}{r_{max}} \tag{14}$$

Where $M(< r) \equiv \int_0^r 4\pi \rho(\tilde{r}) \tilde{r}^2 \mathrm{d}\tilde{r}$ is the mass enclosed by a sphere of radius r such that the clump's total mass is enclosed by the radius r_{max}: $M_{tot} = M(< r_{max})$. The full derivation of equation (14) is given in appendix B.

2.2.3. Accounting for Neighbouring Structures

The boundaries of a particle's trajectory in a given potential ϕ can be estimated using the conservation of energy:

$$E/m_p = \frac{1}{2}v^2 + \phi = const. \tag{15}$$

A gravitational potential is qualitatively shown in figure (3) as well as a particle α with $\frac{1}{2}v^2 < -\phi$. The total energy per particle mass E/m_p of the particle on the graph is then exactly the difference between the negative potential and the kinetic energy on the y-axis. In order to conserve energy, the curve of possible kinetic energies (dotted line in figure 3) that the particle may take on will always have the same distance E/m_p from the negative potential curve. Because $v^2 \geq 0$, the spatial boundaries of a particle's trajectory can be found by following the curve of possible kinetic energies of the particle to the point where $v^2 = 0$.

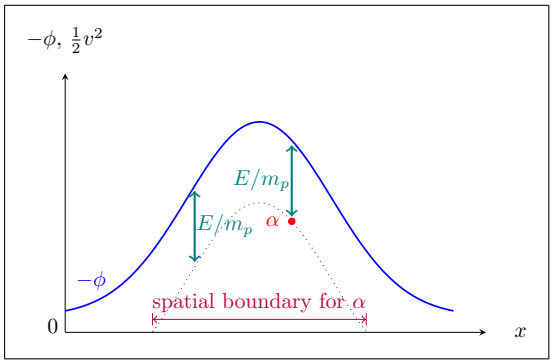

Figure 3: A qualitative plot of a potential $-\phi$ and a particle α. The boundaries of the particle's trajectory can be found using energy conservation $E/m_p = \frac{1}{2}v^2 + \phi$ by following the curve of the particle's kinetic energies (dotted line) to the points where $v^2 = 0$.

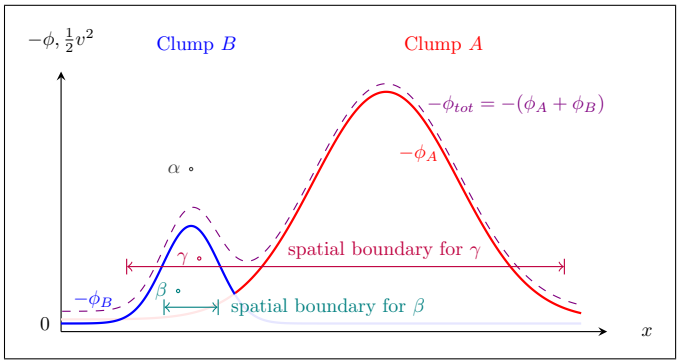

Figure 4: Qualitative potential of a halo containing two clumps A and B. Three particles assigned to B are shown: α is not bound to B, β is bound, γ satisfies the energy condition to be bound, but can wander off into clump A and shouldn't be considered as such.

This fact changes the situation significantly for the interpretation of what particles should be considered bound as follows. Consider now an isolated halo that consists of two clumps, A and B, where B is a smaller clump nested within clump A. Their potentials are qualitatively depicted in figure (4). Three particles assigned to B with different kinetic energies are marked, representing three different cases:

- Particle α has a kinetic energy higher than the potential, it is clearly not bound to the clump B.
- Particle β has a kinetic energy lower than the potential at that distance from the centre of mass, so it will remain bound on an elliptic trajectory around the centre of mass.
- Particle γ is considered energetically bound to the clump just like β, i.e. it satisfies the condition (12), but it won't necessarily remain on an elliptic trajectory around clump B's centre of mass: Because of clump A's neighbouring potential, the particle can leave the boundaries of clump B and wander off deep into clump A.

These considerations show that due to the fact that subhalos will always have neighbouring structure, there will be particles like γ that can wander off into neighbouring clumps even though they satisfy condition (12). It is obvious that particles like γ shouldn't be considered as bound and that therefore the bound condition needs to be modified appropriately. Even though the clumps were assumed to be isolated, clumps that make up substructure will always have at least

one neighbour by their nature. The reason γ can wander off is because its boundary extends past the interface that connects the two clumps, so the condition for a particle to be bound must be that its trajectory must never reach that interface. Defining ϕ_S to be the potential of clump B at the interface to the neighbouring structure that is closest to B's centre of mass, the condition for a particle to be bound *exclusively* to a particular clump can be written as

$$E/m_p = \frac{1}{2}v^2 + \phi - \phi_S < 0 \tag{16}$$

or equivalently:

$$v < \sqrt{-2(\phi - \phi_S)} \tag{17}$$

According to the argumentation above and figures 3 and 4, it is to be expected that demanding particles to be exclusively bound will find more unbound particles than not doing so, where particles close to the centre of mass should be more likely to be exclusively bound than the particles closer to the edge of the subhalo.

2.2.4. Structure Hierarchy: What to Do With Unbound Particles

One last thing that needs to be considered with respect to unbinding particles is the hierarchy of substructures that is found by `PHEW`, as described in section 2.1.2. As mentioned before, for conventional reasons, it is assumed that all particles within a halo are bound to the halo. Therefore there is no need to examine the halo-namegiver clumps for unbound particles.
The situation is different for substructure clumps: If a particle is found to be not bound to a clump, it shouldn't be discarded or removed from the entire structure, but passed on to the next higher level of substructure for examination. This way as much as possible information on substructure is preserved and the unbinding procedure ends naturally when the level of the halo-namegiver clumps is reached.

2.2.5. Biased Clump Properties

The last attempt to improve the unbinding algorithm is to take into account that the properties of the found substructure are biased. Consider this very simplified case: A known spherical halo contains exactly one, also spherical and known, subhalo. Such a situation can be seen in figure 5. By construction, the clumpfinder `PHEW` will not recognise these two as two spherical clumps. Instead, it will group cells by following the steepest density gradient. (The clumps that `PHEW` actually finds can be seen in the top row of fig. 10.) In consequence, the clump properties, e.g. the centre of mass and the bulk velocity, based only on particles that are found to be a part of the clump will differ from the known properties for two reasons:

1. Some of the particles that are known to be a part of the subhalo will be assigned to the halo instead. The found subhalo will have missing particles.
2. The subhalo is nested within the halo. It will be contaminated by the halo's particles.

So in order to retrieve clump properties that are closer to the known properties, some further work needs to be done. It seems likely that the clump properties after particle unbinding should be closer to the known ones, particularly so if neighbouring structures were taken into consideration and therefore the remaining structure contains only exclusively bound particles. A possible improvement is to recompute the clump properties after unbinding and use this

updated information to go through the entire procedure again. The whole unbinding procedure is reiterated until the bulk velocity of each clump converges. The bulk velocity is considered converged when the relative difference of the old and the new bulk velocity is smaller than a user-defined convergence limit ε:

$$\text{bulk velocity converged} \Leftrightarrow \left| \frac{v_{bulk,old} - v_{bulk,new}}{v_{bulk,new}} \right| < \varepsilon \tag{18}$$

This concludes the detailed description of the requirements of the unbinding algorithm. In the following chapter, its implementation is described.

3. The Unbinding Algorithm

The unbinding procedure begins after `PHEW` has identified and merged all clumps. The main outline of the code is as follows:

- Gather all particles that are in test cells that are not halo-namegivers and create a linked list of those particles. Then append the linked lists of substructure to their parent's linked lists.
- Loop over all clump levels starting with the lowest. For each clump of this level:
  - Determine clump properties: centre of mass, clump mass, bulk velocity, particle furthest away from centre of mass
  - Compute the cumulative mass profile by binning the particles of that clump into mass bins
  - If neighbouring structures are considered: Find the shortest distance from each clump's centre of mass to any of its neighbours
  - Calculate the potential for this clump
  - Loop over this clump's particle linked list. If a particle is marked for examination, interpolate the potential at its position and check whether it is bound (condition (17)). If the particle is not bound and iterative clump property determination is not chosen or the bulk velocity of each clump has converged, mark the particle for examination in the clump's parent's linked list. Otherwise, mark it as "not contributing to the clump properties" and repeat the entire loop for this clump.

The listed parts are described in more detail below.

3.1. Particle Gathering and Linked Lists

To gather all particles which are in test cells, a loop over all test cells is performed. Test cells are always leaf cells, but particle linked lists exist only for octs. So in order to identify which particles of the entire oct are in the cell, for each test cell a loop over particle linked list of the oct it belongs to is performed and each particle examined whether it is located within the test cell. If it is, the particle is considered to belong to the clump and the peak label of the test cell is stored for that particle.

For all clumps that are not halo-namegivers, a linked list of all particles that belong to them is created, because halo-namegivers are not considered for particle unbinding. The linked list consists of three arrays: For each peak patch, the first and the last particle of the linked list are stored, so that one can access the linked list at the start and append particles to the list, resp. The third array stores for each particle the particle that follows after it. The linked lists will be used later to loop over only those particles that belong to a particular clump.

Once the loop over all test cells is done, a loop over all clump levels is performed, starting with the lowest level. The linked lists of all clumps of that level are appended to the linked lists of their parent clumps. This is necessary because substructures, even though initially identified as separate structures, are still considered to be part of their parents, and it also enables to pass on unbound particles.

3.2. Determining Clump Properties

With established linked lists of particles of each clump, the properties of clumps can now be computed. Each MPI task computes the total mass, the bulk velocity (eq. (5)) and the centre of mass for all clumps on their respective domain, including the virtual ones, by looping over each clump's particle linked list. The results are then collected with a sum operation across all tasks and the results scattered again to all virtual peaks. (The exact bulk velocity and centre of mass are computed after these communications, because they require the total mass to be known.)
If the clump properties are supposed to be determined iteratively, only particles marked as "contributing" will contribute to the properties. Initially, all particles contribute and they are marked as "not contributing" in the unbinding step. After the clump properties are computed, all particles of the clump are marked as contributing again, so they can be unbound anew.
In case the simulation uses periodic boundary conditions, a correction for every dimension individually is applied to particle with distances greater than half of the box length by shifting their position by a half of the box length closer, where the box length is the size of the total computational domain.
Furthermore, with known centres of mass, the distance of the particle furthest away from the centre of mass is determined by looping over the particle linked lists again. Just like before each MPI task computes the maximal distance of the particles on its domain to their respective centres of mass. The results are stored, communicated across tasks, the maximal value for each clump kept and scattered to the corresponding virtual peaks. From this point on, whether a clump should be considered for examination will be determined by whether the distance of the particle furthest away from the clump's centre of mass is greater than zero. It is a suitable condition because the value for said distance is initiated to zero and communicated to all virtual peaks, but it also ignores all clumps that consist of only one particle[1]: It doesn't really make sense to examine whether a particle is bound to itself.

3.3. Determining the Cumulative Mass Profile

To compute each clump's cumulative mass profile, the distance from the centre of mass to the particle furthest away is divided into bins. The number of bins is a parameter that can be set manually. It also can be chosen whether to use linear or logarithmic bin distances. Linear distances are evenly spaced, so the outer boundary r_i (with respect to the centre of mass) of the $i-$th bin can be obtained with

$$r_i = i \cdot \frac{r_{max}}{n_{bins}} \tag{19}$$

where r_{max} is the distance of the particle furthest away from the clump's centre of mass and n_{bins} is the number of bins used.

[1] Even with a condition for a clump's minimal mass to be greater than one particle's mass, the clump does not necessarily contain more than one particle. Due to the CIC interpolation scheme, the density of a cell also depends on the particles in neighbouring cells. As a result, a cell can satisfy the overdensity condition without actually containing enough particles itself.

The bin widths for logarithmic binning distances grow exponentially in size with increasing distance from the centre of mass. The outer boundary r_i of the $i-$th bin can be obtained with:

$$r_i = r_{min} \cdot \left(\frac{r_{max}}{r_{min}} \right)^{\frac{i}{n_{bins}}} \tag{20}$$

The first bin is set to start at the distance r_{min}:

$$r_{min} = \frac{\text{boxlen}}{2^{\text{levelmax}}} \tag{21}$$

where `levelmax` is the deepest level of mesh refinement throughout the simulation and `boxlen` is the size of the entire computational domain. r_{min} is chosen this way to represent the simulation's resolution.
Again a loop over the particle linked lists is performed and the particle's mass deposited in the bin corresponding to their distance to the centre of mass of the clump they belong to. The masses in bins are summed with a collective communication and the result scattered across to the virtual peaks, yielding each clump's mass profile. Then each clump's mass bins are summed up starting from the bin closest to the centre of mass, resulting in cumulative mass profiles.

Note that all particles assigned to a clump, even those previously marked as "not contributing" or as belonging to a substructure on a lower level, always contribute to the cumulative mass profile. The mass profile nevertheless needs to be recomputed if clump properties are determined iteratively, because the centre of mass is expected to change with every iteration.

3.4. Finding the Closest Saddle

In order to account for neighbouring structures, for each clump, the closest point from its centre of mass to any neighbouring clump must be found. This point is also called the "closest saddle". To find the closest saddle, a loop over all test cells is performed. For each cell, all its neighbouring cells are gathered on the level of the cell, of a level above and the level below (if it exists). Then all neighbouring cells that are not leaf cells are discarded, because only leaf cells make up the peak patch cells. Furthermore, all neighbouring cells are ignored if they are either in the same clump as the test cells under investigation or in no clump at all. Then the connecting point between the test cell and its neighbour is calculated and its distance to the clump's centre of mass computed. The minimal distance is stored for each clump and communicated across MPI tasks.
The routines to find the closest saddle are slightly modified routines that are used already in `PHEW`.

3.5. Computing the Potential

Each clump's gravitational potential is calculated according to eq. (14). The previously computed cumulative mass profile of the clump gives the enclosed mass $M(< r)$ needed for the integral. The integral is computed for every bin distance r_i. The precise value of the potential at each particle's position will later be interpolated by using the computed values of the two closest bin distances.

First, the integral between each two neighbouring bins is computed and stored by using a simple

trapezoidal rule:

$$\int_{r_i}^{r_{i+1}} \mathrm{d}r \frac{GM(<r)}{r^2} \approx (r_{i+1} - r_i) \cdot \frac{1}{2} \left(\frac{GM(<r_{i+1})}{r_{i+1}^2} + \frac{GM(<r_i)}{r_i^2} \right) \tag{22}$$

Then the potential can be inferred with

$$\phi(r_i) = \sum_{j=i}^{N} \left(- \int_{r_j}^{r_{j+1}} \mathrm{d}r \frac{GM(<r)}{r^2} \right) - \frac{GM_{tot}}{r_N} \tag{23}$$

3.6. Unbinding Particles

In order to do unbind particles, a loop over the clump's particle linked list is performed and each particle examined whether it satisfies condition (17). To determine the potential ϕ a particle experiences, it first needs to be determined "in which mass bin the particle is located". Once that is known, the potential that the particle experiences is inferred with a linear interpolation. Suppose a particle with distance d from the centre of mass of the clump is "in the bin" i, which is to say $r_{i-1} < d < r_i$, where r_i is the $i-$th cumulative mass binning distance. Since the potential at every binning distance r_i has been calculated, the potential $\phi(d)$ that the particle experiences is interpolated with:

$$\phi(d) = \left(\frac{\phi(r_i) - \phi(r_{i-1})}{r_i - r_{i-1}} \right) (d - r_{i-1}) + \phi(r_{i-1}) \tag{24}$$

The potential at the position of the closest saddle, ϕ_S, is also calculated this way.

If a particle is found to be unbound (doesn't satisfy condition (17)), one of two following things will happen. If iterative clump properties determination is chosen, unbound particles will be marked as "not contributing". After all particles of clumps of this level have been examined, the entire loop is repeated, starting with the determination of clump properties. This procedure is repeated until the bulk velocities of all clumps of this level are converged (or reach a manually set upper limit for the number of iterations). Clumps whose bulk velocity is converged are marked as such and not re-iterated over. Once the clump properties are converged, the actual removal of the unbound particles from clumps begins. If iterative clump properties determination is not selected, the code just skips the iteration part and jumps to this last step.
If a particle is not bound, its peak label is changed to the clump's parent's peak label. Having the same peak label as the clump under investigation marks a particle for examination: The only way a particle can acquire a different peak label than the clump in whose linked list it is in is to be assigned to a substructure on a lower level first. Looping over the clump levels starting with the lowest level makes sure that each particle that is not assigned to a halo-namegiver is examined at least once.

4. Test Cases

To test the unbinding procedure, the program will be executed on three datasets.

The first two of these datasets, named "`dice-twobody`" and "`dice-levels`", contain highly idealistic structures, where the effects of particle unbinding and the determination of the clump's

properties can be seen and evaluated more easily. They are both created with `DICE` (Perret 2016), which "*models initial conditions of idealized galaxies [...]. The code can set up a large number of components modelling distinct parts of the galaxy, and creates 3D distributions of particles*" (Perret 2016). Each structure is initially generated as an isolated, spherically symmetric halo. The halos were chosen to follow the Navarro-Frenk-White (NFW) mass profile. The individual halos are later joined together to create one large structure that contains substructure(s). All particle masses are set to be identical. Furthermore, all particles of a clump are set to be energetically bound to the clump, thus satisfying eq. (12), where the potential ϕ is computed the same way as given in eq. (14).

The `dice-twobody` dataset consists of two clumps created this way. A smaller halo, made of 40'000 particles, is nested within a bigger halo that contains 200'000 particles, so that it represents a subhalo. Additionally, the subhalo is set to follow a Kepler orbit with respect to the bigger one with the eccentricity $\varepsilon = 0.1$ so that it has a closed orbit. The initial particle distribution of the two halos is shown in figure 5.
The `dice-twobody` dataset will be used to demonstrate the effects of the choice of parameters relevant for the unbinding procedure in chapter 5, which are the number of mass bins used to compute the potential and the convergence limit ε for iterative clump properties determination.

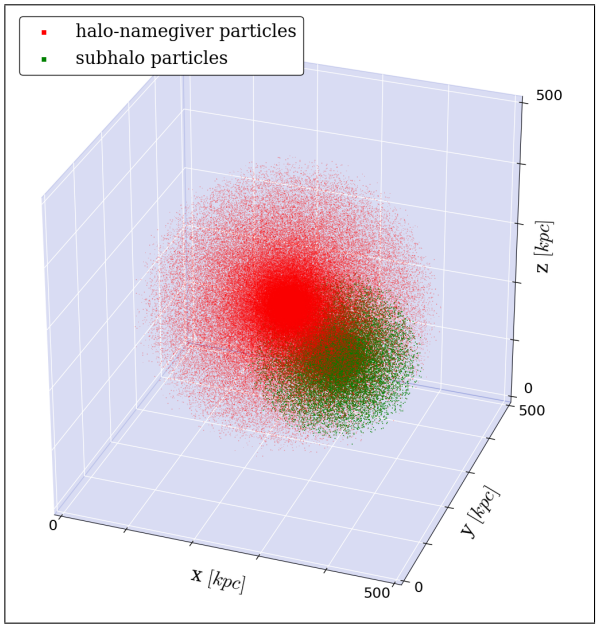

Figure 5: The initial particle distribution of the `dice-twobody` dataset. A smaller halo (subhalo 1) made of 40'000 particles is nested within a bigger halo (halo-namegiver), which contains 200'000 particles.

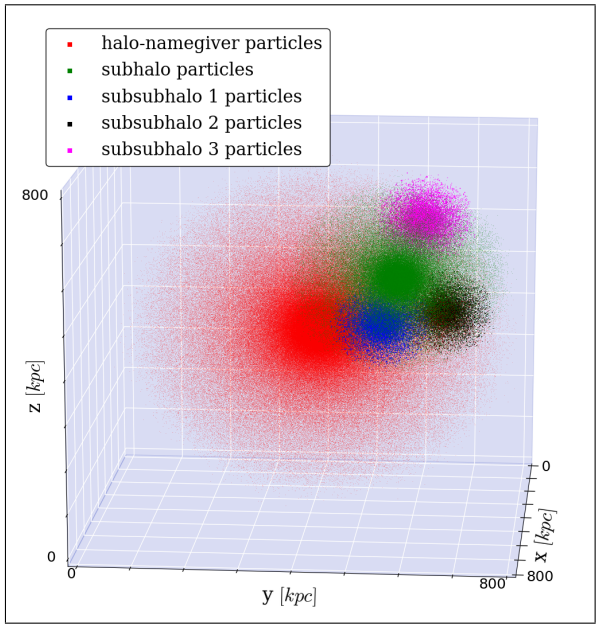

Figure 6: The initial particle distribution of the `dice-levels` dataset. 3 small halos (subsubhalo 1-3), made of 20'000 particles, are around a bigger halo (subhalo 1), which contains 200'000 particles. These four structures are then enveloped by an even bigger halo (halo-namegiver), made of 1'000'000 particles.

The second test case created with `DICE`, the `dice-levels` dataset, consists of a halo made of 200'000 particles that has three subhalos, each containing 20'000 particles, following close Kepler orbits around it. Then this composite structure is enveloped by an even bigger halo with 1'000'000 particles. The purpose of this dataset is mainly to demonstrate the particle unbinding on a multi-levelled structure hierarchy. The initial particle distribution is shown in figure 6.

The third dataset, named `cosmo`, is a cosmological simulation, aiming to simulate very (physically) large structures and their formation over a long time interval. As such, it differs in

multiple aspects to the previous two. Firstly, the particles are described with comoving coordinates to account for the expansion of the Universe. The computational domain is a cube ("*box*") of fixed length 1 that contains all the particles. The physical coordinates of the particles at a certain time can then be computed with respect to the expansion scale factor at that time, thus implementing the expansion of the Universe without need to actually resize the box or move the particles at each time step solely because of the expansion. Secondly, the simulation uses periodic boundary conditions. This means that if a particle would pass a surface of the box, it is reintroduced at the opposite surface. The usage of periodic boundary conditions allows to simulate an infinite Universe by considering only a part of it and interpreting this part as representative of the entire Universe.

The `cosmo` dataset consists of 128^3 ($\approx 2 \cdot 10^6$) dark matter particles at redshift $z = 0$ with the Hubble constant $H_0 = 70.4$ and density parameters $\Omega_m = 0.272$ and $\Omega_\Lambda = 0.728$ [2]. The density threshold for clump finding was chosen to be 80 times the cosmological critical density ρ_{crit} and the saddle threshold for halos was set to $200\rho_{crit}$. Only halos with more that 10 particles were kept. The particle distribution is shown in figure 7.

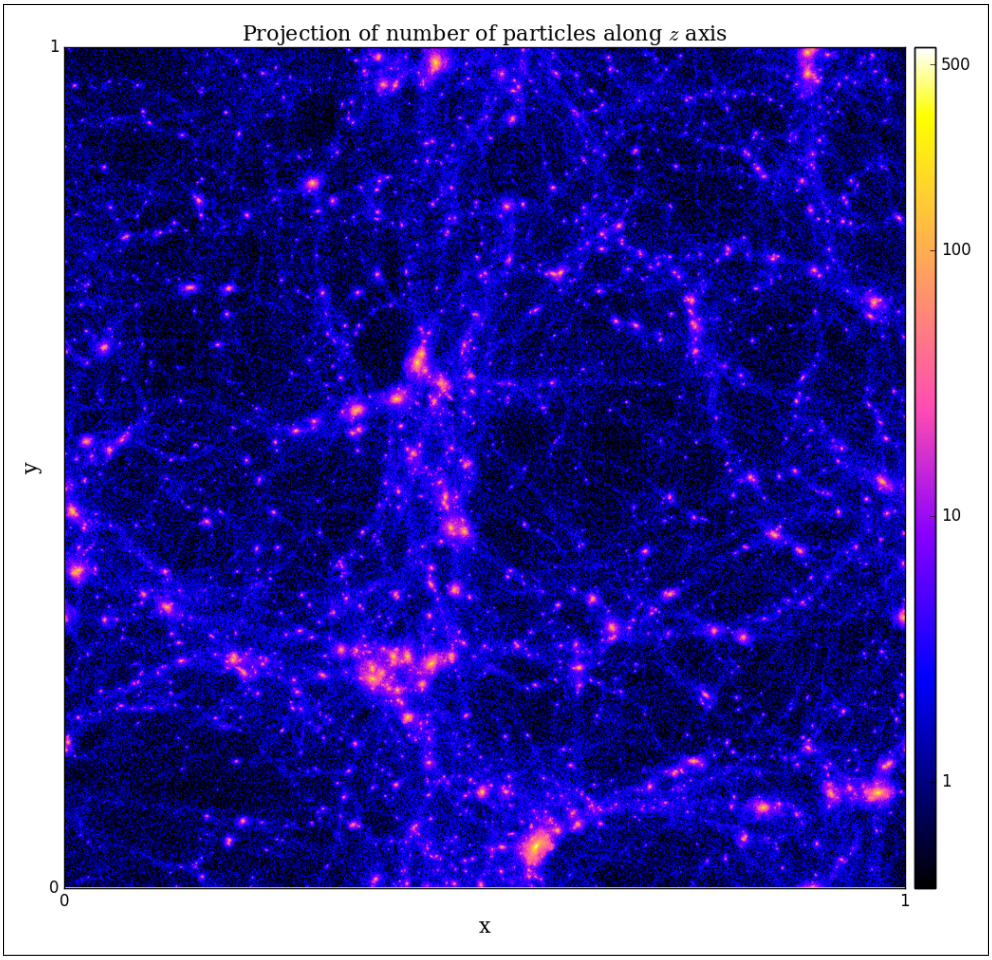

Figure 7: The particle distribution of the `cosmo` dataset. It is a cosmological simulation of 128^3 dark matter particles at redshift $z = 0$ with $H_0 = 70.4$ and density parameters $\Omega_m = 0.272$ and $\Omega_\Lambda = 0.728$.

[2] The density parameter of some cosmological component i is defined as $\Omega_i = \frac{\rho_i}{\rho_{crit}}$, where ρ_i is its density and ρ_{crit} is defined as $\rho_{crit} = \frac{3H^2}{8\pi G}$, where H is the Hubble parameter and G is the gravitational constant.

5. Results

5.1. Accuracy of the Potential in Dependance of the Number of Mass Bins

Fig. 8 shows potentials computed with different numbers of mass bins, ranging from 10 to 10'000, and different binning distances (linear or logarithmic). The potentials are computed for the subhalo of the `dice-twobody`-dataset. The results of 10'000 bins were taken as a reference point, which contain ~ 5 particles per bin on average. Figure 9 shows the standard deviation σ_N and the maximal deviation $D_{max,N}$ of the potential obtained with N bins computed with

$$\sigma_N = \left[\frac{1}{N} \sum_i \left(\frac{\phi(r_i)_N}{\phi(r_i)_{10'000}} - 1 \right)^2 \right]^{\frac{1}{2}} \tag{25}$$

$$D_{max,N} = \max_i \left\{ \left| \frac{\phi(r_i)_N}{\phi(r_i)_{10'000}} - 1 \right| \right\} \tag{26}$$

where r_{max} is the distance of the particle furthest away from the centre of mass ($r_{CoM} = 0$).

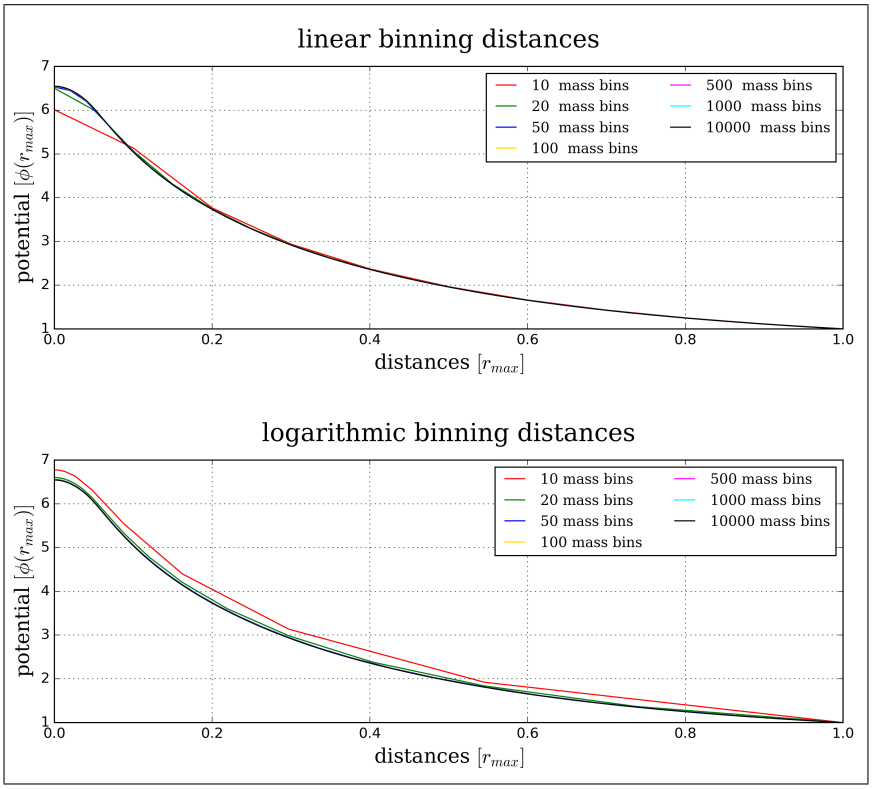

Figure 8: The potential of the subhalo clump of the `dice-twobody`-dataset computed with different binning methods and bin numbers. r_{max} is the distance of the particle furthest away from the centre of mass, which is set at $r = 0$

For both linear and logarithmic binning distances, the maximal deviation falls well below 1% for 50 mass bins used, and below 0.1% for 100 mass bins. Even though the standard deviation of linear binning is smaller for 100 bins, logarithmic bins are used for further tests because the logarithmic binning seems to describe the potential close to the centre of mass better. This is particularly important when the unbinding includes iterative clump properties determination and considers neighbours, which will be shown in section 5.3.

Notably, the potential obtained with 10 bins and logarithmic binning distances seems to be the

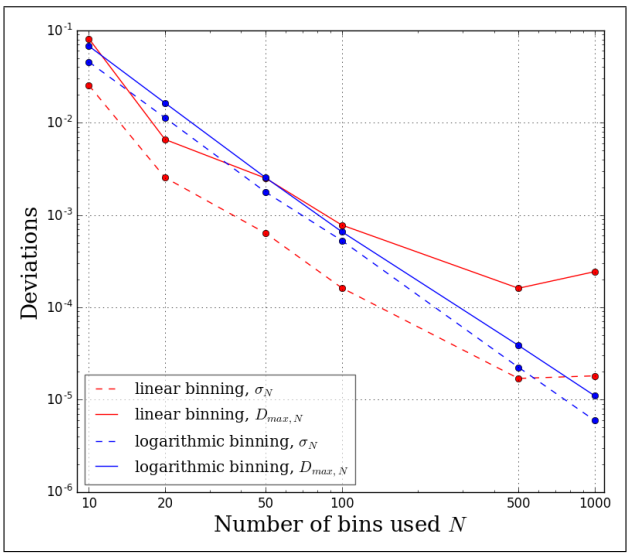

Figure 9: Standard deviation σ_N and maximal deviation $D_{max,N}$ of the potential obtained with N bins, computed according to equations 25 and 26, resp., for the subhalo clump of the `dice-twobody`-dataset

only one deviating on every point. This is caused by the way the integral for the potential is computed: As the bin widths grow for logarithmic binning distances, the accuracy decreases. The error for 10 bins and logarithmic binning distances should be the highest of all. Once the integral of each bin is computed, the integral is then summed up beginning at the outermost bin with respect to the centre of mass, so a large error at the first bin will propagate through every other bin.

5.2. Effect of Iterative Clump Properties Determination

ε	d_v	`niter`	D_v
0.5	0.3000	2	0.2326
0.1	0.0614	4	0.0419
0.01	0.0036	7	0.0024
0.001	0.0010	8	0.0014
0.0001	0.0000	10	0.0009

Table 1: The deviation D_v (eq. 27) from the originally set bulk velocity to the computed bulk velocity for the subhalo of the `dice-twobody`-dataset in dependence of the convergence limit ε. The iterative clump properties determination halts after `niter` iterations, when the difference d_v of the previously determined bulk velocity to the new bulk velocity, $d_v = \frac{v_{bulk} - v_{bulk,old}}{v_{bulk,old}} < \varepsilon$.

For the `dice-twobody`-dataset, the precise relative velocity of the two clumps is known. Table 1 shows the deviation D_v of the computed clump bulk velocity v_{bulk} from the original bulk velocity v_{orig} as set with `DICE` with decreasing converging limits ε. D_v is computed as follows:

$$D_v = \left| \frac{v_{bulk} - v_{orig}}{v_{orig}} \right| \tag{27}$$

The deviation falls below 1% for $\varepsilon = 0.01$, which is why that value will be used for further tests.

5.3. Effects of Different Unbinding Methods

Taking neighbouring structures into account and determining clump properties iteratively are attempts to improve the accuracy of the particle unbinding, but can be left out if desired. In this section, the results of three different unbinding methods on the three previously introduced datasets as well as the results as found by the clump finder without unbinding. The methods are named as follows:

`simple` Simple unbinding without considering neighbour clumps for the potential nor determining clump properties iteratively.

`neighbours` Unbinding considering neighbour clumps for the potential

`iter` Unbinding considering neighbour clumps for the potential and iterative clump properties determination

All three unbinding methods are executed with 100 mass bins and logarithmic binning distances. For the `iter` runs, the convergence limit ε was set to 0.01. The results without particle unbinding, which are the results as found by `PHEW`, are also the initial situation on which the unbinding is performed. They are labelled as "`PHEW only`".

Table 2 shows the number of particles in clumps for these unbinding methods. For the `cosmo`-dataset, all halo-namegiver and subhalo particles have been summed up respectively, because there were too many clumps ($\sim 19'000$) to represent individually. Furthermore, the particle distributions for all four runs and all three datasets are shown in figures 10, 11 and 12. In case of the `cosmo`-dataset, only one halo and its substructure are shown instead all of them simultaneously, so the effects of the unbinding methods can be seen better.

	dice-twobody-dataset			
	PHEW only	simple	neighbours	iter
halo-namegiver particles	187606	202998	233450	231905
subhalo particles	52394	37002	6550	8095

	dice-levels-dataset			
	PHEW only	simple	neighbours	iter
halo-namegiver particles	965485	975038	1196614	1180544
subhalo particles	197767	221977	49647	58963
subsubhalo 1 particles	38734	23398	4475	7280
subsubhalo 2 particles	26591	19590	5239	6151
subsubhalo 3 particles	31421	19995	4023	7060

	cosmo-dataset			
	PHEW only	simple	neighbours	iter
halo-namegiver particles	573832	640686	746441	740616
subhalo particles	188036	121182	15427	21252

Table 2: The number of particles in clumps depending on the unbinding method for the `dice-twobody`-, `dice-levels`- and `cosmo`-dataset.

Comparing the results of `PHEW` and the `simple` unbinding of the `dice-twobody`-dataset in figure 10a and of the `cosmo`-dataset in figure 12a shows that the `simple` method already has a decent

contribution to the assignment of particles to the structures. Looking at the halo-namegiver particles only, one can see that where PHEW has made clean cuts between the clumps, now particles have been re-introduced, yielding smoother structures.

The results of the simple unbinding for the dice-levels-dataset in fig. 11a and table 2 show clearly that most unbound particles from the lowest level substructure, the subsubhalos, have been assigned to the parent clump, not to the halo-namegiver. Note for example that most substructure particles above the subsubhalo 1 (green) are assigned to the subhalo (pink) after simple unbinding. The surfaces of the subsubhalos facing away from the subhalo appear much more spherical, which is how they were originally created to be.

In all three figures, the effect of considering neighbours for the potential seems drastic. But recall that when neighbouring clumps are considered for the potential, the particles that remain after unbinding are exclusively bound to the subhalo they are assigned to. The substructure clumps have indeed been stripped from most of their particles and only what appears to be "cores" close to the centre of mass remain, in agreement with what one would expect considering the argumentation used in context with figures 3 and 4. Two subhalos from the halo of the cosmo-dataset (figure 12, subhalo 1 and 5, have even been stripped of all their particles, indicating that none of their particles can be considered as exclusively bound to them.

The iterative determination of clump properties seems to find overall more bound particles compared to the neighbours runs. In each of the presented cases the total number of particles in substructure grew. In case of the cosmo-dataset, 36 clumps had less bound particles (472 particles in total) after iter-unbinding, while 187 clumps had more bound particles, 6297 in total.

Unfortunately, iterative clump properties determination couldn't re-introduce any particles to the completely unbound subhalos 1 and 5 of the halo shown in fig. 12b. If all particles are unbound at the first iteration already, no new clump properties can be determined and the situation remains as it is. If necessary, an attempt to improve on this effect might be to first determine clump properties iteratively for such cases and only after the iteration has finished take neighbouring structures into account. This feature has not been adapted, as no urgent need for it was seen. It is perfectly plausible that some substructure as identified by PHEW could contain no exclusively bound particles.

5.4. Resource usage

It was not possible to measure the resource usage of the unbinding algorithm in isolation, because it only works within the frame of RAMSES and PHEW. That is why the CPU time and the peak memory consumption per core (PMCPC) were measured not only with the unbinding procedure executed ("unbinding runs"), but also when the clump finding was performed without unbinding ("PHEW runs") and runs without clump finding at all ("RAMSES runs"). Each run consist of advancing the simulation for one time step. The clump finding and unbinding procedures, when used in a run, are always called twice: First after the initialisation, then after the advance in time of the simulation.

The measurements were made using the GNU time version 1.7 command on an Intel Core i7-6700HQ processor with four cores. For the two datasets created with DICE, both sequential runs, performed on only one core, as well as parallel runs, performed on four cores (without hyper-threading) were measured. Measurements of sequential runs of the cosmo-dataset were not possible, because of the lack of a dataset that can be run on only one core. The "unbinding

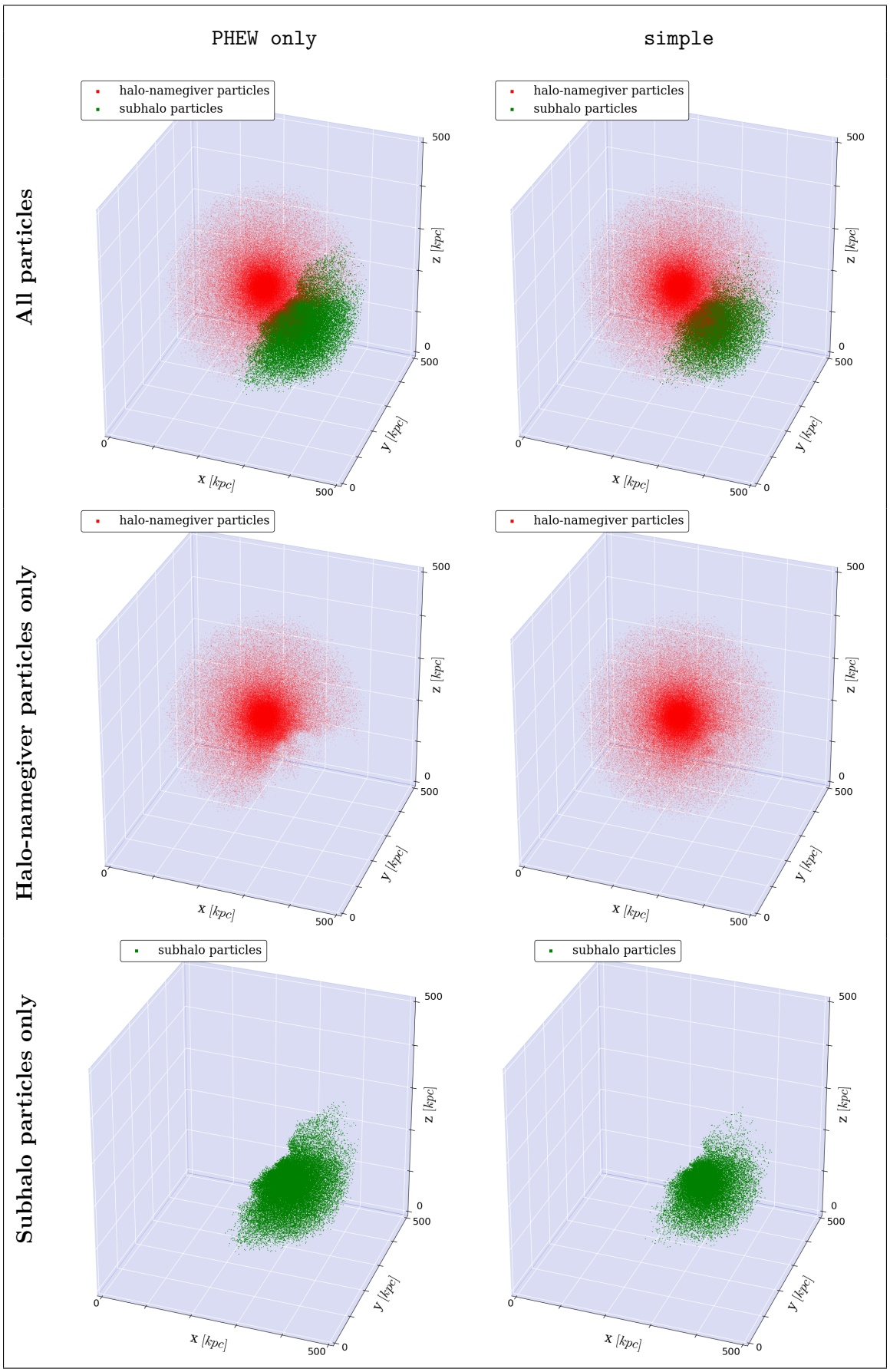

Figure 10a: The results of `PHEW only` and `simple` unbinding of the `dice-twobody`-dataset: All particles, halo-namegiver particles only and subhalo particles only.

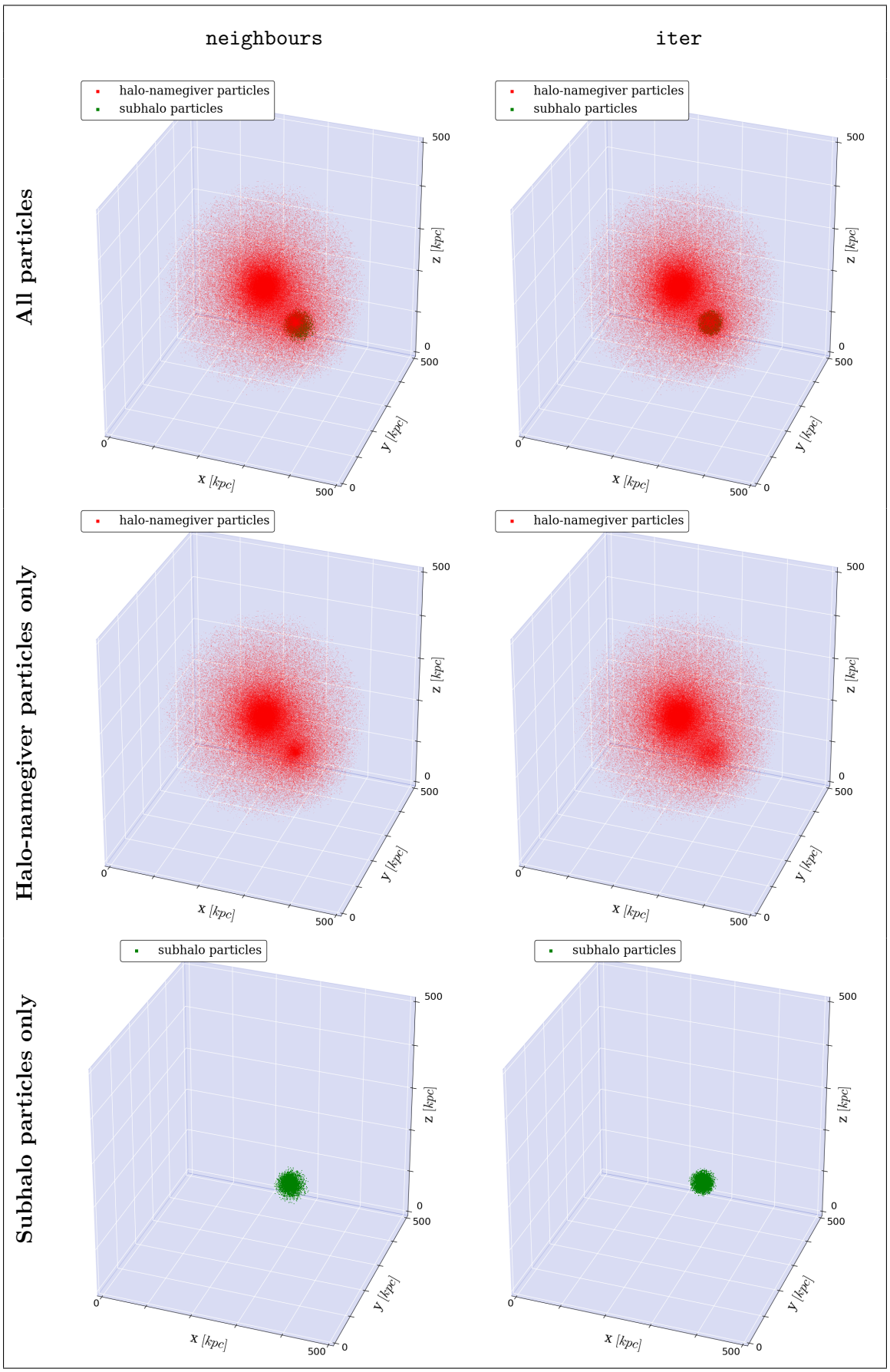

Figure 10b: The results of `neighbours` and `iter` unbinding of the `dice-twobody`-dataset: All particles, halo-namegiver particles only and subhalo particles only.

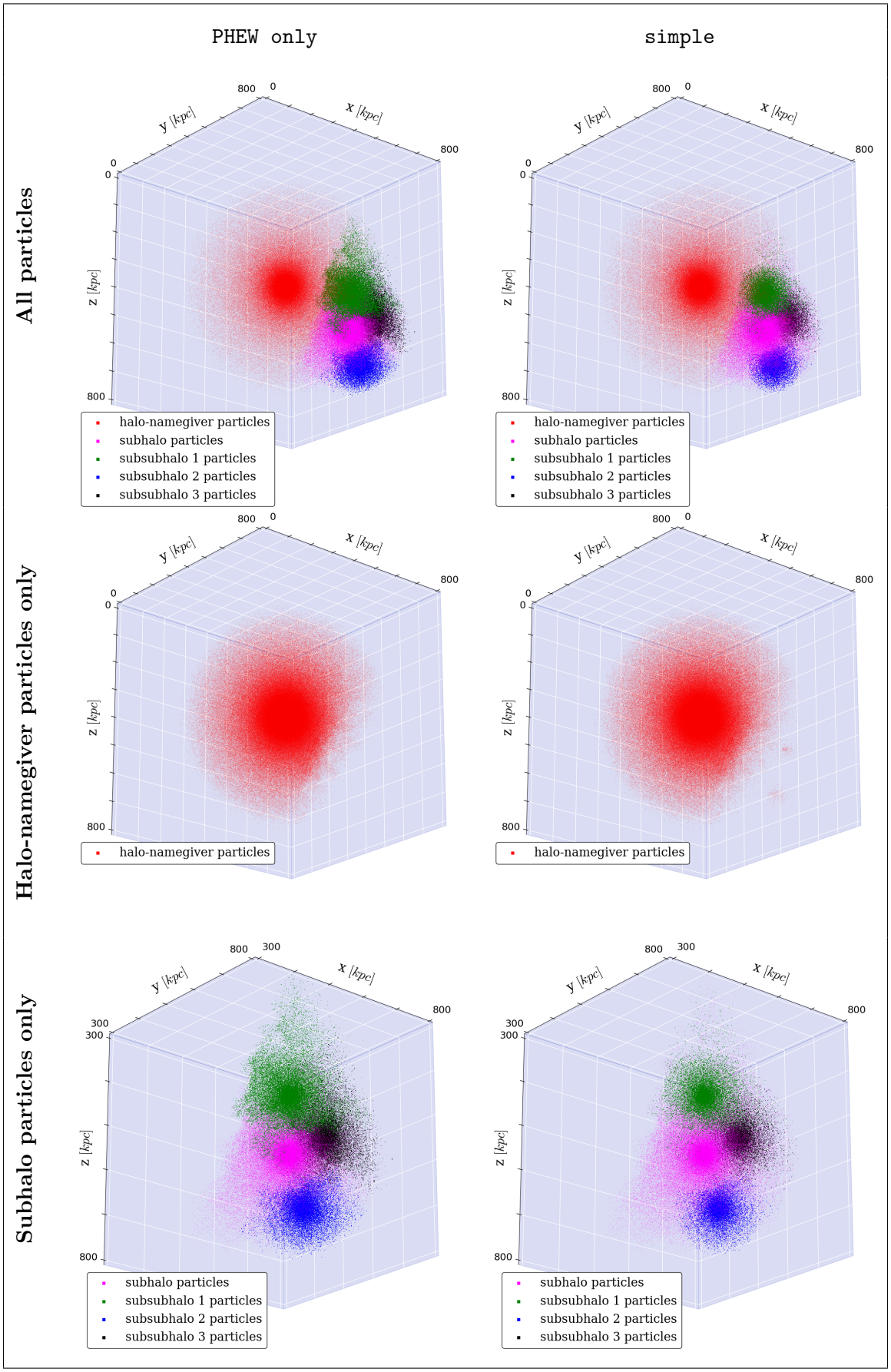

Figure 11a: The results of `PHEW only` and `simple` unbinding of the `dice-levels`-dataset: All particles, halo-namegiver particles only and subhalo particles only.

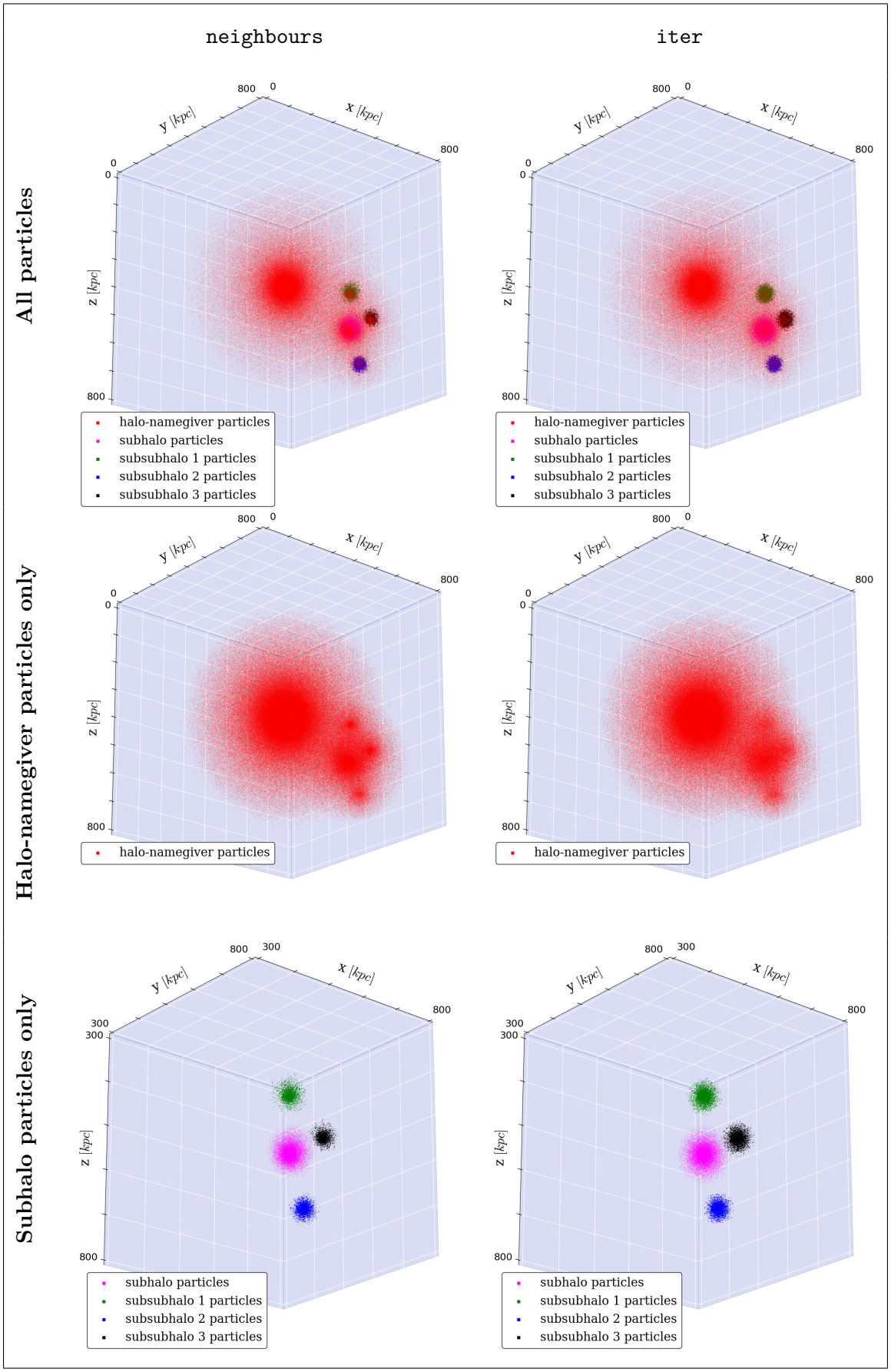

Figure 11b: The results of `neighbours` and `iter` unbinding of the `dice-levels`-dataset: All particles, halo-namegiver particles only and subhalo particles only.

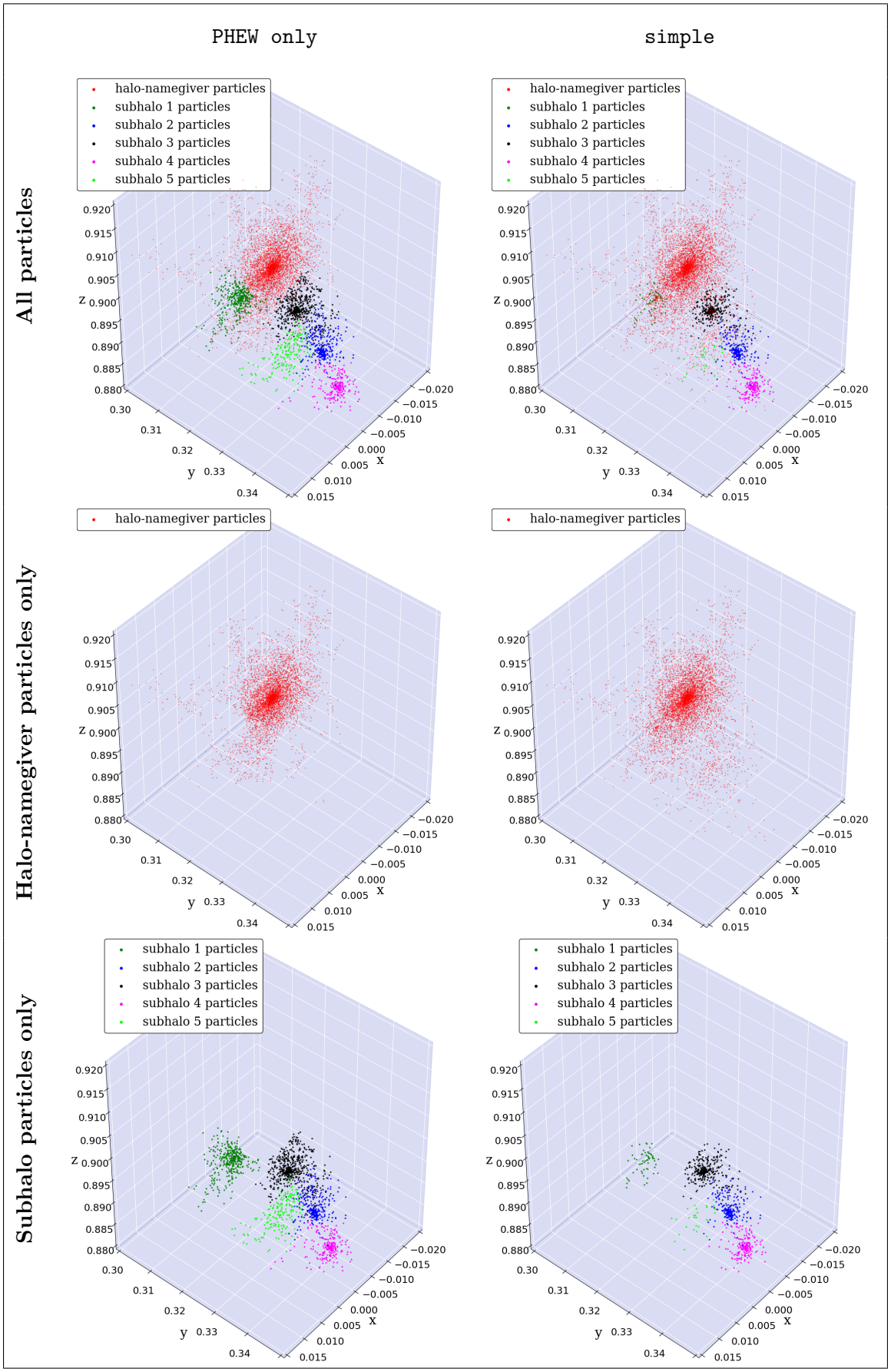

Figure 12a: The results of `PHEW only` and `simple` unbinding of the `cosmo`-dataset: All particles, halo-namegiver particles only and subhalo particles only.

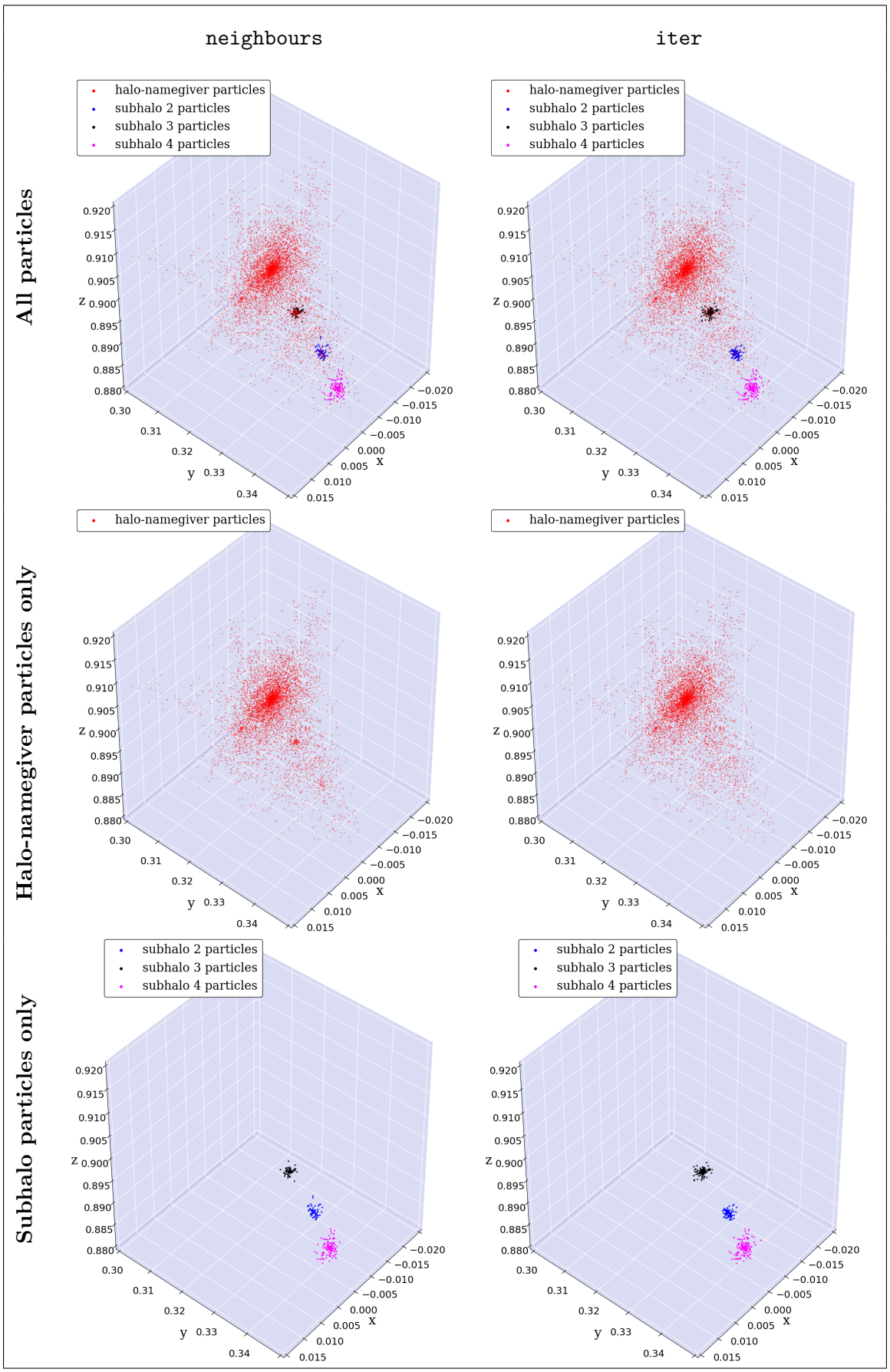

Figure 12b: The results of `neighbours` and `iter` unbinding of the `cosmo`-dataset: All particles, halo-namegiver particles only and subhalo particles only.

	dice-twobody-dataset ($2.4 \cdot 10^5$ particles, 21.8% of which are in subhalos)					
	Sequential execution			Parallel execution		
	RAMSES	PHEW	unbinding	RAMSES	PHEW	unbinding
CPU time (s)	16.18 +34.8 %	19.83 +10.0 %	21.81 -	26.06 +85.3 %	35.71 +35.3 %	48.30 -
PMUPC (Mb)	390.50 +15.8 %	401.37 +12.7 %	452.37 -	396.76 +44.3 %	551.09 +3.9 %	572.54 -

	dice-levels-dataset ($1.26 \cdot 10^6$ particles, 23.4% of which are in subhalos)					
	Sequential execution			Parallel execution		
	RAMSES	PHEW	unbinding	RAMSES	PHEW	unbinding
CPU time (s)	56.77 +55.7 %	82.31 +7.4 %	88.41 -	281.60 +58.0 %	361.30 +23.1 %	444.93 -
PMUPC (Mb)	1313.36 +18.9 %	1358.45 +15.0 %	1562.06 -	405.96 +54.1 %	565.26 +10.7 %	625.64 -

	cosmo-dataset (128^3 particles, 9.0% of which are in subhalos)					
	Sequential execution			Parallel execution		
	RAMSES	PHEW	unbinding	RAMSES	PHEW	unbinding
CPU time (s)	-	-	-	383.72 +16.9 %	432.93 +3.6 %	448.46 -
PMUPC (Mb)	-	-	-	1335.61 +14.3 %	1342.77 +13.7 %	1526.65 -

Table 3: Performance measurements for sequential and parallel executions on the three datasets used for tests. The parallel runs were executed on 4 cores. The CPU time of the parallel executions is the total CPU time of all 4 cores. "*PMUPC*" stands for "peak memory usage per core". "RAMSES" measurements consist of advancing the simulation for a timestep. "PHEW" runs do the same, but include clump finding, while "unbinding" runs include clump finding and particle unbinding. The given values are the highest values out of 10 measurements. The percentages below the measurements give the relative additional cost of the unbinding run.

runs" are set up identically to `iter`-runs because this method requires the most resources. 100 mass bins and iterative clump properties determination with a convergence limit ε of 0.01 were used. Each run was repeated ten times, out of which the highest resource usage is shown in table 3.

These measurements are mostly meant as a guideline for future users. Very little comparisons can be made between the three datasets, because the necessary resources depend on a multitude of factors, such as particle numbers, how far the mesh will be refined, the number of particles that will end up being in halos and subhalos and so forth, some of which are very difficult to control.

The additional computational cost is expected to be strongly affected by the number of particles that are in subhalos, as these are the ones that the unbinding procedure works on. Particles in halos are not considered for unbinding. Comparing the CPU times of the `dice-levels`-dataset with the times of the `cosmo`-dataset supports this expectation. Even though the `cosmo`-dataset contains ~ 1.7 as many total particles as the `dice-levels`-dataset, it has ~ 1.6 times less particles that are actually in subhalos. If the number of particles in subhalos wasn't an important factor for the computational cost, the CPU times of the runs including the unbinding procedure shouldn't be nearly identical.

The measurements also indicate that the relative additional cost compared to the runs without

unbinding decreases with increasing total particle number. This may be explained by the fact that most particles that are found to be in a structure will be in halos, thus won't contribute significantly to the cost of the unbinding procedure, while raising the expenses for `RAMSES` and `PHEW`. This interpretation is supported in the case of the two datasets created by `DICE`, where the `dice-levels`-dataset contains more particles in subhalos both in absolute number and percentage, but requires noticeably less relative additional resources.

The additional required resources seem acceptable for a code designed to work on-the-fly, particularly so in the case of the `cosmo`-dataset. If required, the execution time might be accelerated by vectorising the code, which for most parts hasn't been done yet.

6. Conclusion

A particle unbinding algorithm for the halo-finder `PHEW` of the N-body simulation code `RAMSES` was introduced and the effects of three different unbinding methods were demonstrated (table 2), which are:

1. simple unbinding of particles which don't satisfy an energy condition (right column of fig. 10a, 11a and 12a)
2. exclusive unbinding, considering that particles can wander off into neighbouring structures (left column of fig. 10b, 11b and 12b), and
3. exclusive unbinding, where additionally the structure properties are determined iteratively using the remaining (bound) particles (right column of fig. 10b, 11b and 12b).

The simple unbinding was already able to smooth out parent structures, which were previously cut off cleanly by `PHEW` and in artificially set up, highly idealistic cases, both halo and substructures partially regained their original form.
Exclusive unbinding removed most of the substructure particles, leaving only a "core". This method is clearly not suitable to preserve most of substructure information, but could be useful to track the evolution of substructure through time, which is currently not possible with `PHEW`[3].
The iterative determination of clump properties leads to overall less unbound particles in each case, but still most particles were unbound. In some cases like shown in fig. 12, subhalos were found to have no exclusively bound particles. The iterative clump properties determination of the code as presented in this thesis can't improve such situations. If required, it might be worth considering first to determine the clump properties of these subhalos iteratively and only then applying the correction for neighbouring structures.
What might also further improve the results would be a better way to calculate the gravitational potential that the particles experience. Currently, the unbinding algorithm introduces the assumption of spherical symmetry to calculate the potential, thus partly ignores the shape of the clumps identified by `PHEW`.

The algorithm works on-the-fly and is fully parallel, using the MPI library. Its additional resource usage (table 3) seems acceptable, particularly so with decreasing relative amount of particles that are assigned to substructure. If necessary, the code might be sped up by fully vectorising it.

[3] This has not been tested yet and might be subject of future work.

Acknowledgements

I want to thank Prof. Teyssier for his guidance and support for this thesis; I want to express my gratitude to the lovely people from the ICS for their help in various matters, in particular to Dr. Andreas Bleuler and Dr. Valentin Perret; And I want to thank my family and friends for their support, love and patience.

References

[1] P. Behroozi, R. Wechsler, and H.-Y. Wu. *Rockstar: Phase-space halo finder*. Astrophysics Source Code Library. Oct. 2012. ascl: 1210.008.

[2] M. J. Berger and P. Colella. "Local adaptive mesh refinement for shock hydrodynamics". In: *Journal of Computational Physics* 82 (May 1989), pp. 64–84. DOI: 10.1016/0021-9991(89)90035-1.

[3] Andreas Bleuler et al. "PHEW: a parallel segmentation algorithm for three-dimensional AMR datasets". In: *Computational Astrophysics and Cosmology* 2.1 (2015), pp. 1–16. ISSN: 2197-7909. DOI: 10.1186/s40668-015-0009-7. URL: http://dx.doi.org/10.1186/s40668-015-0009-7.

[4] M. Davis et al. "The evolution of large-scale structure in a universe dominated by cold dark matter". In: *The Astrophysical Journal* 292 (May 1985), pp. 371–394. DOI: 10.1086/163168.

[5] A.M Khokhlov. "Fully Threaded Tree Algorithms for Adaptive Refinement Fluid Dynamics Simulations". In: *J. Comput. Phys.* 143.2 (July 1998), pp. 519–543. ISSN: 0021-9991. DOI: 10.1006/jcph.1998.9998. URL: http://dx.doi.org/10.1006/jcph.1998.9998.

[6] A. Knebe et al. "Haloes gone MAD: The Halo-Finder Comparison Project". In: *Monthly Notices of the RAS* 415 (Aug. 2011), pp. 2293–2318. DOI: 10.1111/j.1365-2966.2011.18858.x. arXiv: 1104.0949.

[7] A. Knebe et al. "Structure finding in cosmological simulations: the state of affairs". In: *Monthly Notices of the RAS* 435 (Oct. 2013), pp. 1618–1658. DOI: 10.1093/mnras/stt1403. arXiv: 1304.0585.

[8] S. R. Knollmann and A. Knebe. "AHF: Amiga's Halo Finder". In: *The Astrophysical Journal* 182 (June 2009), pp. 608–624. DOI: 10.1088/0067-0049/182/2/608. arXiv: 0904.3662.

[9] V. Perret. *DICE: Disk Initial Conditions Environment*. Astrophysics Source Code Library. July 2016. ascl: 1607.002.

[10] Planelles, S. and Quilis, V. "ASOHF: a new adaptive spherical overdensity halo finder". In: *Astronomy and Astrophysics* 519 (2010), A94. DOI: 10.1051/0004-6361/201014214. URL: https://doi.org/10.1051/0004-6361/201014214.

[11] W. H. Press and P. Schechter. "Formation of Galaxies and Clusters of Galaxies by Self-Similar Gravitational Condensation". In: *The Astrophysical Journal* 187 (Feb. 1974), pp. 425–438. DOI: 10.1086/152650.

[12] V. Springel et al. "Populating a cluster of galaxies - I. Results at [formmu2]z=0". In: *Monthly Notices of the RAS* 328 (Dec. 2001), pp. 726–750. DOI: 10.1046/j.1365-8711.2001.04912.x. eprint: astro-ph/0012055.

[13] J. G. Stadel. "Cosmological N-body simulations and their analysis". PhD thesis. UNIVERSITY OF WASHINGTON, 2001.

[14] Teyssier, R. “Cosmological hydrodynamics with adaptive mesh refinement. A new high resolution code called RAMSES”. In: *Astronomy and Astrophysics* 385 (Apr. 2002), pp. 337–364. DOI: 10.1051/0004-6361:20011817. eprint: astro-ph/0111367.

Appendix A Glossary

closest saddle	The closest point from a clump's centre of mass to any interface with any neighbouring clump.
clump	A peak patch that satisfies the relevance condition; A group of particles.
halo	A gravitationally bound cosmological object.
halo-namegiver	A relevant clump that will never be merged into another clump, while other clumps may be merged into it. Its peak label will be the label of the entire halo.
key neighbour	The neighbour of a peak patch that is connected by the key saddle.
key saddle	The point of the saddle surface between two peak patches with the highest average density.
leaf cells	Cells which are not (further) refined
oct	Basic elements of the data structure in `RAMSES`. A group of 2^{ndim} cells of same size.
parent clump	The clump a clump will be merged into.
peak	A local density maximum.
peak label	An unique index assigned to each peak to identify it.
peak patch	Spatial section made of test cells that have been assigned to a particular peak.
saddle	The surface between two neighbouring peak patches
subhalo	A gravitationally bound object within a halo.
task (MPI)	A unit of execution, e.g. a processor.
test cells	Leaf cells which have a density above an user-defined density threshold.

Appendix B Solution of the Poisson Equation for a Spherically Symmetric Case

Consider a spherically symmetric clump of radius r_{max} and total mass M_{tot} such that $\rho(r > r_{max}) = 0$ and therefore $M_{tot} = \int_0^{r_{max}} \rho(r)\mathrm{d}r$. Furthermore, suppose that the first derivative $\frac{\partial\phi}{\partial r}$ exists at the point $r = 0$ and chose $\phi(r \to \infty) \equiv 0$ as a point of reference.
The spherically symmetric Poisson equation can be written as:

$$\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial\phi}{\partial r}\right) = 4\pi G\rho(r) \tag{B.1}$$

Integrating the equation once with respect to dr from 0 to r gives

$$r^2\frac{\partial\phi}{\partial r} - \underbrace{\left[r^2\frac{\partial\phi}{\partial r}\right]_{r=0}}_{=0} = 4\pi G\int_0^r \rho(\tilde{r})\tilde{r}^2\mathrm{d}\tilde{r} = GM(<r) \tag{B.2}$$

Where $M(<r) = \int_0^r 4\pi\rho(\tilde{r})\tilde{r}^2 \mathrm{d}\tilde{r}$ is the mass enclosed by a sphere of radius r such that $M_{tot} = M(<r_{max})$. Equation B.2 shows that demanding $\frac{\partial\phi}{\partial r}|_{r=0}$ to be finite can be justified physically as $\frac{\partial\phi}{\partial r} \propto M(<r)$ and one expects the enclosed mass $M(<r) \to 0$ for $r \to 0$.

That leaves us with:

$$\frac{\partial\phi}{\partial r} = \frac{GM(<r)}{r^2} \tag{B.3}$$

$$\Rightarrow \ \phi(r) = G\int_0^r \frac{GM(<\tilde{r})}{\tilde{r}^2}\mathrm{d}\tilde{r} + \phi_0 \tag{B.4}$$

An expression for ϕ_0 is found by using the boundary condition $\phi(r \to \infty) = 0$:

$$\phi(r\to\infty) = G\int_0^\infty \frac{M(<\tilde{r})}{\tilde{r}^2}\mathrm{d}\tilde{r} + \phi_0 \tag{B.5}$$

$$= G\int_0^{r_{max}} \frac{M(<\tilde{r})}{\tilde{r}^2}\mathrm{d}\tilde{r} + G\int_{r_{max}}^\infty \frac{M(<\tilde{r})}{\tilde{r}^2}\mathrm{d}\tilde{r} + \phi_0 \tag{B.6}$$

$$= G\int_0^{r_{max}} \frac{M(<\tilde{r})}{\tilde{r}^2}\mathrm{d}\tilde{r} + G\frac{M_{tot}}{r_{max}} + \phi_0 \tag{B.7}$$

$$= 0 \tag{B.8}$$

$$\Rightarrow \phi_0 = -G\int_0^{r_{max}} \frac{M(<\tilde{r})}{\tilde{r}^2}\mathrm{d}\tilde{r} - G\frac{M_{tot}}{r_{max}} \tag{B.9}$$

For the second integral in equation (B.6) it was used that for all $r \geq r_{max}$ the enclosed mass remains constant: $M(<r)|_{r\geq r_{max}} = M_{tot}$:

$$\int_{r_{max}}^\infty \frac{M(<\tilde{r})}{\tilde{r}^2}\mathrm{d}\tilde{r} = \int_{r_{max}}^\infty \frac{M_{tot}}{\tilde{r}^2}\mathrm{d}\tilde{r} = M_{tot}\left[-\frac{1}{\tilde{r}}\right]_{\tilde{r}=r_{max}}^{\tilde{r}=\infty} = \frac{M_{tot}}{r_{max}} \tag{B.10}$$

Plugging equation (B.9) into (B.4) yields:

$$\phi(r) = G\int_0^r \frac{GM(<\tilde{r})}{\tilde{r}^2}\mathrm{d}\tilde{r} - G\int_0^{r_{max}} \frac{M(<\tilde{r})}{\tilde{r}^2}\mathrm{d}\tilde{r} - G\frac{M_{tot}}{r_{max}} \tag{B.11}$$

$$= -G\left(\int_0^{r_{max}} \frac{M(<\tilde{r})}{\tilde{r}^2}\mathrm{d}\tilde{r} - \int_0^r \frac{M(<\tilde{r})}{\tilde{r}^2}\mathrm{d}\tilde{r}\right) - G\frac{M_{tot}}{r_{max}} \tag{B.12}$$

$$= -G\int_r^{r_{max}} \frac{M(<\tilde{r})}{\tilde{r}^2}\mathrm{d}\tilde{r} - G\frac{M_{tot}}{r_{max}} \tag{B.13}$$