Hierarchical Clustering and Galaxy Cluster Scaling Laws

by

Eric Robert Tittley

Graduate Programme

in

Astronomy

Submitted in partial fulfilment

of the requirements for the degree of

Doctor of Philosophy

Faculty of Graduate Studies

The University of Western Ontario

London, Ontario

December 1998

© Eric Robert Tittley 1999

National Library of Canada
Acquisitions and Bibliographic Services
395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale du Canada
Acquisitions et services bibliographiques
395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-40291-6

Canada

Abstract

The dependency of the morphology of the hot gaseous component of clusters of galaxies on the hierarchical nature of the structure formation in our universe is examined. The indicators of morphology examined are the universal density profile of Navarro, Frenk, and White (1995), the mass-temperature relationship, and the biasing of the gas with respect to the dark matter. The study is done using high resolution numerical simulations of clusters of galaxies. The simulations model the collisionless (dark matter) component as well as the baryonic matter using an N-body code with smoothed particle hydrodynamics (SPH) parallelised using Pthreads. The clusters are evolved from initial conditions smoothed by top-hat filtering and low-pass filtering of the initial perturbation spectrum. The evolution takes place in a $40h^{-1}\,\mathrm{Mpc}$ volume.

The mean dark matter density profiles from each of the models is found to be fit well by the universal profile. A discontinuous form described in the text finds $\rho \propto r^{-1.8}$ in the inner regimes of the clusters, independent of the model. The density in the outer regimes is found to depend on the degree of smoothing, becoming more shallow with increased smoothing. The mass-temperature relation is found to depend on the initial conditions, as well. All models reproduce the $T \propto M^{2/3}$ relation, but the coefficient of proportionality is found to decrease with increased smoothing of the initial conditions. This is traced to an increase in the isothermal radius of the clusters. The gas in the clusters is found to be anti-biased with respect to the dark matter. This anti-bias is reduced with smoothing of the initial conditions. In particular for the clusters formed hierarchically from unsmoothed initial conditions, there is a strong positive bias in the outer radii of the clusters.

A description of the method of parallelisation is given as well as results of tests

of SPH involving cooling near a steep density gradient and the drag on a cold clump moving through a hot media. The tests are done for a variety of implementations of SPH which vary both the method of symmetrising the equations of motion and the form of the artificial viscosity. Both of these are found to not have significant effects.

KEYWORDS: Clusters of Galaxies, Cosmology, Smoothed Particle Hydrodynamics, Parallel Programming, Universal Profile, Mass-Temperature Relation, Baryon Fraction.

Table of Contents

List of Tables

List of Figures

Chapter 1

INTRODUCTION

1.1 Clusters of Galaxies

The largest virialised objects in the universe are galaxy clusters. Larger than this is the regime of large scale structure. Optically, these clusters were initially identified from maps of galaxies which indicate galaxies tend to be found in clusters. Subsequently, these clusters of galaxies have been found to contain a diffuse intergalactic medium composed of hot ($10^8\,\mathrm{K} \approx 10\,\mathrm{keV}$) gas. This gas is luminous ($\sim 10^{11}\,\mathrm{L}_\odot$) in the x-ray regime and, consequently, observable from above the earth's atmosphere. Two such satellites in particular, *Einstein* and ROSAT, have provided catalogues of X-ray luminosities as well as luminosity maps with sufficient resolution to map the coarse structure of these clusters. This hot gas constitutes the bulk of the baryonic matter in galaxy clusters.

Satellites with instruments able to take spectral data in the X-ray band, most notably the ASCA satellite, have provided observations which have confirmed the high temperatures of the gas and are starting to coarsely resolve the spatial temperature distribution of this gas (see Markevitch and Vikhlinin (1997a), for references).

Since these clusters are such large objects, in the standard model of cosmological

structure formation they are the most recently formed large-scale objects. Consequently, though they are relaxed, they potentially carry with them information related to their formation. Larger-scale structure is in a state of linear or quasi-linear evolution and, hence, still responding to density perturbations that have existed since early times. Smaller virialised objects such as galaxies and stars individually contain little or no manifestation of the character of the initial density perturbations from which they formed. Having had ample time to dynamically relax, their present evolution is dominated by the physics of stellar evolution.

Being so luminous, galaxy clusters are observable to great distances and, consequently, at earlier stages of their evolution. Galaxy clusters are observable in ways other than their X-ray signature and population of galaxies. In the microwave spectrum, there is the temperature decrement imposed on the cosmic microwave background via the Sunyaev-Zeldovich effect by which electrons in the hot gas alter the energy spectrum of the CMB photons. Optical imaging of the clusters also provides independent measurements of the total mass of the clusters via the gravitational lensing of background galaxies.

1.2 A Universal Density Profile

The work of Navarro, Frenk, and White (1995, 1996) has built a case for a common density profile for the dark matter in galaxy clusters. This density profile is found to fit clusters spanning a large range of masses. It is contended that this 'universal' profile has only one free parameter corresponding to the density at cluster formation (Navarro, Frenk, and White 1997). However, the form of the profile is derived from numerical simulations of hierarchically formed clusters. There is evidence that it is the hierarchical nature of the cluster formation itself that is responsible for the universal profile (Syer and White 1998). Determining the validity of this would be valuable to understanding the true breadth of its universality.

1.3 The Mass-Temperature Scaling Relation

Estimations of the total mass of the clusters have been made using the observed X-ray luminosity (and hence, baryon) distribution and the assumption of hydrostatic equilibrium using so-called β-model fits (Fabricant and Gorenstein 1983; Jones and Forman 1984). Evrard, Metzler, and Navarro (1996) describes a correlation between the total mass of clusters of galaxies and the mass-weighted mean temperature of the hot, X-ray emitting gas in the interior of the cluster. This would provide an independent method of measuring the total masses of the clusters, requiring only a measurement of the temperature of the gas. This relationship is a natural result of the hydrostatic state of the gas. However, derivation of this relationship uncovers a dependency on both the gas density and temperature profiles. Hence, a sensitivity of these profiles to the cosmogony would weaken the relation's utility. Understanding the behaviour of the mass-temperature scaling relation on the cosmogony is essential to the confidence level put in the application of the relation to real data.

1.4 Cluster Baryon Fractions

Being the largest virialised objects in the universe, galaxy clusters contain matter sampled from volumes with radii of $\simeq 8h^{-1}\,\mathrm{Mpc}$. Presuming that the gas and dark matter was initially distributed in equal proportions, these samples remain unbiased. Since galaxy clusters are the largest objects for which total mass may be measured, the baryon fraction of the universe, *i.e.* the total mass of baryons to the total of all mass, predicted by primordial nucleosynthesis calculations may be directly compared to that of this large and presumably unbiased sample of matter. However, there is an uncertainty in the actual amount of bias. The bias between the universal baryon fraction and that found in clusters cannot be measured directly. It must be inferred from numerical simulations. The numerical simulations must assume a model for the universe. Since the actual cosmogony of the universe is unknown in the details,

variation among baryon fractions found in different model universes adds uncertainty. A better understanding of how this bias depends on cosmological models would assist in understanding the actual baryon fraction of the universe.

1.5 Numerical Simulations of Galaxy Clusters

Numerical simulations of galaxy clusters using more than a few hundred particles date back to the work of Efstathiou and Eastwood (1981). These simulations, and many that followed, traced the evolution of the dark matter component only. Despite the omission of the more computationally expensive gas component, they still provide fertile datasets due to their ability to go to very high spatial resolutions (Moore et al. 1998; T.G. Brainerd 1998; Thomas et al. 1998). Numerical simulations of clusters incorporating gas were being performed by the late 1980's (Evrard 1988; Evrard 1990). Many of these used the smoothed particle hydrodynamics (SPH) approximation. SPH is a Monte Carlo approximation which models the gas as a collection of particles with their individual properties taken to be the average of their immediate neighbour particles. The early simulations involved small numbers of particles (2×16^3, typically) which poorly resolved the hydrodynamic forces. Higher spatial resolutions were achieved by modelling single clusters (Thomas and Couchman 1992; Tsai, Katz, and Bertschinger 1994) and then extrapolating to the general case.

More significantly, improvements in codes and computing power continuously push the resolutions to new highs. Improvements benefit both the spatial and temporal resolution, with the former being the more important owing to the large dynamic range of scales in cosmological scenarios. One of the methods by which codes and computing power have increased in step is in the development of algorithms that take advantage of particular advancements in architecture. Software that utilises the ability of some hardware platforms to concurrently perform multiple tasks is one example. Writing code from scratch to do this is possible, but often conversion of

well-tested codes already in existence is required. For the example given, this is referred to as 'parallelisation' of the code. To take advantage of the dual processor UltraSPARCs available, the N-body code with hydrodynamics, HYDRA(Couchman, Thomas, and Pearce 1995), was parallelised.

With improvements in the resolution of numerical simulations has come a need to improve the implemented approximations of true physics as well as to incorporate new physics. This is particularly true for the evolution of the baryonic component. As part of a series of tests of SPH, the cooling rate of the gas around a cold, dense object as well as the hydrodynamic drag on a similar object as it passes through a hot medium were examined for changes owing to differing implementations of SPH.

1.6 The Standard Model

The model of the Universe assumed in this thesis is predicated on the Big Bang. That is, at early times, the Universe was much denser and hotter. A general expansion of the Universe has led to the lower mean-density state of the present. Many excellent reviews of this topic exist. See Bertschinger (1994), for example.

In the Standard Model, the early Universe is assumed to be smooth in both density and temperature with small fluctuations on all scales. These small fluctuations owed their existence to some unknown process from an even earlier time, such as quantum fluctuations during a postulated inflationary period. Before the epoch of recombination, during which the density and temperature dropped sufficiently to allow photons to travel essentially unimpeded, growth of these density perturbations was inhibited. After recombination, these fluctuations grew due to gravitational instability. Initially, they grow in a linear or quasi-linear phase, during which the amplitude of the fluctuations grow at a similar rate to the expansion factor of the universe, a. Eventually the perturbations reach a non-linear phase in which turbulence, angular momentum, or some other factor slows or halts the growth of these fluctuations. This non-linear

phase is highly chaotic. Perturbations with different initial amplitudes reach this stage at different times, with the largest initial perturbations going non-linear first.

There exists a great deal of evidence for the Standard Model, and the Big Bang, in general. This evidence includes the expansion of the Universe as quantified by Hubble's Law as well as the ability of primordial nucleosynthesis calculations to justify the present observed abundances of hydrogen and helium as well as the relative absence of the heavier metals. The Cosmic Microwave Background, via its thermal nature as well as its small spatial fluctuations, provides strong support for both the higher initial temperatures and densities as well as the existence of the initial small-amplitude density fluctuations.

These features are typical of models based on the Big Bang. Added to the Standard Model is the assumption that the dominant source of gravitational mass is an unseen 'Dark' component. Evidence includes the flat rotation curves of galaxies, the dynamics of galaxies in galaxy clusters, as well as the hot intra-cluster medium described earlier.

For both theoretical (*e.g.* inflation) and aesthetic reasons (*e.g.* we should not be living in a preferred time), the Universe is assumed to have a mean density that is critical in the sense that it is just sufficient to halt the expansion of the Universe, albeit after an infinitely long time. The ratio of the density to this critical density, as given by the parameter Ω, is unity with the dark matter contribution, Ω_{DM}, providing the bulk.

1.7 Hierarchical Clustering

In hierarchical clustering, the largest structures forming at a given time do so via the amalgamation of many smaller structures which have formed at an earlier time. This is owing to the form of the initial density perturbation spectrum in which small

scale perturbations have higher initial amplitudes than large-scale. In contrast, non-hierarchical clustering involves structure formation from the collapse of large structures with smooth density distributions. Though the details are still not clear, the results of numerical simulations compared with observations support the theory that we live in a universe in which structure is formed hierarchically. The degree to which the hierarchical nature affects galaxy clusters is not entirely clear. In this thesis, a comparison of the features of galaxy clusters discussed above will be made using numerical simulations of galaxy clusters formed hierarchically and non-hierarchically. In particular, the issue of whether the mass-temperature scaling relationship and the common density profile are universal among both hierarchical and non-hierarchical cluster formation scenarios is explored, as well as the change in baryon biasing within the clusters. By using these extreme cases, the significance of hierarchical clustering itself will be determined. The analysis will concentrate not on the properties of individual clusters, but on the global mean properties of scaled quantities.

1.8 Layout of Thesis

The layout of the thesis is as follows. In Chapter 2, the parallelisation of HYDRA is described as well as results concerning the performance gain. In Chapter 3, SPH is briefly described with emphasis on the differences between the implementations used in the tests. An examination of the behaviour of the SPH density estimate in the presence of steep density gradients is given, followed by the results of the cooling and drag tests. In Chapter 4, the simulations are described. The analysis and, in particular, the cluster identification method are described in Chapter 5. Verification that the clusters extracted are in hydrostatic equilibrium is made in Chapter 6. The results of the examination of the dark matter density profiles are given in Chapter 7. The mass-temperature relationship relevant to clusters is derived in Chapter 8 and the results are given. Chapter 9 discusses the variation of the biasing of the baryons

within the clusters. A brief recap of the results is given in Chapter 10. The casual reader could skip Chapters 2, 3, 5, and 6.

Chapter 2

PARALLELISATION OF HYDRA

2.1 Introduction

With the increasing availability of inexpensive symmetric multiprocessor (SMP) computational resources which use shared memory (Intel x86 machines running Linux or Windows-NT, Sun SPARC-family computers running Solaris, and machines using DEC Alpha CPUs, for example), there is reason to adapt the N-body code, HYDRA (Couchman, Thomas, and Pearce 1995) (see Chapter 4), to take advantage of these architectures. This is possible by parallelising those parts of the code which are the run-time bottlenecks. Parallelisation of code can be done using the propriety libraries supplied with many system libraries. SGI provides the MIPSPro FORTRAN 77 instruction set for use with their multiprocessor systems. HYDRA has been ported to the SGI Power Challenge and Origin 2000 systems using the parallelisation routines provided in the MIPSPro library (Thacker, Couchman, and Pearce 1998) as well as to the Cray using CRAFT (Pearce and Couchman 1997), a FORTRAN-like language for the Crays. Sun Microsystems provide a similar set, as well. Parallelised code written for one system is not transferable to other systems using different operating systems and architectures. Though leading to possible performance degradation, the versatility provided by a platform-independent method would be most welcome.

This platform independence is provided by the POSIX threads library (pthreads). POSIX (Portable Operating System Interface) provides a set of IEEE standards by which writers of system libraries for various architectures may maintain compatibility of software written for other architectures. The downside of the pthreads comes from it being a C library. HYDRA is written in FORTRAN 77, which does not easily mesh with C code. There is no cross-platform FORTRAN 77 threads library freely available at the moment. There are commercial FORTRAN 77 compilers with library extensions for threading, such as APR's FORGExplorer family of FORTRAN 77 tools. OpenMP, a recently announced shared-memory API standard, is scheduled to be available sometime in 1998.

2.2 An Introduction to Parallelisation

Simply put, a program is parallelised if it can use more than one CPU at a time. The CPUs, or in more general terms, the **processing elements** (PEs), obviously must be part of a system in which the processors can be coordinated in some fashion. This system is called a **multiprocessor** system. These processors may be on the same motherboard or on different continents. They may communicate and coordinate with each other or be coordinated by an extra processor.

How the PEs communicate with each other and with their memory determines the type of multiprocessor system which, in turn, determines the method by which the programme is coded and/or compiled to take advantage of the multiple processors.

If the CPUs share the same memory, as they are apt to do if they are on the same motherboard, then the system is said to have a **shared memory** (SM) architecture. If each CPU has its own memory, as many super-computers do, the system is said to have **distributed memory** (DM).

If the processors have equal access to shared memory and are being controlled by

the same operating system then the system is a **symmetric multiprocessor** (SMP) computer. This is the architecture of interest here. The number of processors on the inexpensive systems described earlier is typically 2 to 8.

Compilers exist which will parallelise code to the best of their ability if they are asked to by a compiler directive. This is known as **implicit parallelisation.** For best performance, it is most often the case that the code itself must be modified and/or annotated to direct the compiler at compiler time and the scheduler at run time how to run the code in parallel. This is **explicit parallelisation.**

The minimum requirement for code to be parallelisable is that it have code chunks that can occur in any order. Say the code does some sequence of events, A-B-C-D-E. Now swap any pair of events, such as A-B-D-C-E. If this operation does not change the result of the sequence, then the code may be parallelisable. If no such pair exists, then the code is not parallelisable. Of course, it may be possible in this case to rewrite the code into a new sequence, G-H-I-J-K, that does fit the criterion.

From this description, it can be seen that the parallelisability of a code may be tested on a serial (one CPU) computer. Noting that the code chunks C-D are interchangeable in their sequence, let us rewrite the code in the manner A-B-(C-D)-E where the brackets indicate that the contained code chunks may be processed in any order or even incrementally for each. The scheduling of the code execution is left to an external source, such as the operating system. It is sometimes beneficial to write code for a parallel architecture even if the code is to be run on serial machine. If one code chunk is held up by an I/O delay, C for example, it may be suspended while the other chunk, D, is processed.

It is often the case that the code sequences which can occur in any order modify the same resource such as a variable in memory. Consider as an example the summation

of results from a pair of mathematical operations. The summation can occur in any order, but the counter variable is common to both mathematical operations. Using our previous sequence, suppose C calculates a value x_1 which it adds to x and D calculates a value x_2 which it also adds to x. There is no difference to the final value of x if x_2 is added before x_1. A problem can occur, however, if they both increment x at the same time. The processor for C grabs the value of x from memory, adds x_1 to it, and returns the new value of x to memory. If the processor for D grabs the value of x from memory after the processor for C but before that processor has returned the value to memory, then D will overwrite the memory location for x with the initial value for x incremented by x_2. The incrementation done by C's CPU and all of C's work will have been lost. This simultaneous need to update a memory location in shared memory occurs when there is a **race condition**. A race condition does not guarantee that the simultaneous accesses will occur. It merely states that it *could* occur. Any code written to work in parallel must be prepared to avoid this potential problem.

2.3 Introduction to Threads

Now we turn our attention to the method to be used here for creating parallel code: threads.

Threads make it possible for a process to be split into subtasks which may be executed independently of one another by the operating system's scheduler. Each thread has its own stack and set of registers (memory resources) yet maintains access to the common memory of the process spawning the thread. As well, each thread maintains the same process identifier (PID). This is in contrast to the traditional UNIX method of spawning separate processes with the `fork` system call which creates identical copies of the parent process, except for the PID and return values. This is wasteful of memory and entails expensive overhead.

In practice, a thread is a subroutine that is called in a fashion that indicates to the operating system that the subroutine may be scheduled for processing by an available PE independently of the calling process. Along with this ability, of course, are library routines to collect and synchronise the threads.

The thread may be allowed to run until it terminates quietly on its own, perhaps when it has finished some lengthy writing to disk. This is a **detached thread**. During this time, the process can continue on with other things and no further synchronisation is required.

More often, the main programme wishes to get the results of the thread. If the routine cannot go on without the results, there is little to be gained by spawning a thread for the subroutine. Indeed, there will be a degradation of performance. However, if the code is parallelisable (see the previous section) then there will be a collection of subroutines for which the calling routine requires results but not in any particular order. In this case, the subroutines may be spawned off as threads. The calling routine then waits until all the threads have finished, collects the results, and continues on. If there is more than one PE, then the threads may run at the same time on different PEs, in parallel. In principal, if there are N PEs, $M \times N$ threads, and each thread takes the same length of time to run, then the speed-up through this section of the code should be N times.

However, it is never possible to fully parallelise a code. There always exists a certain amount of the code, contributing the fraction s to the total execution time, which cannot be parallelised. The potential speedup of the code is then bounded by $\frac{1}{s}$. This is Amdahl's Law. As an example, take the following derivation: Allow s to be the fraction of total execution time spent doing tasks which must be done serially. Similarly, take p to be the total fraction of time taken on a serial machine doing tasks which may be done in parallel. If f is the ratio of the total time spent on a serial

machine to that spent on a parallel machine with N processors, then f is the speedup factor. So,

$$f \equiv \frac{t_s}{t_p} \leq \frac{s+p}{s+\frac{p}{N}} \tag{2.1}$$

states Amdahl's Law. In the limit $N \to \infty$, the maximum speedup is bound by the fraction of the code that is un-parallelisable, $f = \frac{1}{s}$.

2.4 Threading FORTRAN 77 Code Using Pthreads

FORTRAN 77 does not support threads. There exist propriety extensions to FORTRAN which support threads but they are machine-dependant. Pthreads provides a machine-independent and inexpensive, if not free, method to thread code, but it does not support FORTRAN as Pthreads comes as a C-based library.

In order to use Pthreads with FORTRAN, a wrapper function must be created. This wrapper function is C-based and thus can use the pthreads tools to create threads of the FORTRAN code that is to be run in parallel.

Consider the example schematically illustrated in Chart 2.1. Here, a programme has a section which contains a loop for which the order of indexing is unimportant. The loop itself may be isolated as a separate subroutines (Chart 2.2). Once isolated, a wrapper function may be created which calls the subroutine more than once. The subroutine's parameters may be adjusted on each call to ensure that the successive calls to the subroutine span non-overlapping ranges of the index, `i`, while ensuring that the entire span of `i=1 to N` is covered (Chart 2.3). Since the order of execution is unimportant, the successive calls can be in any order, or the calls can be all at the same time, using threads.

Let us look at a less schematic example. Consider the programme fragment in Prog. 2.1. It contains just such a parallelisable loop as in Chart 2.1. With our

Chart 2.1 A programme with a parallelisable section. The section may be a series of commands or a loop, as it is here.

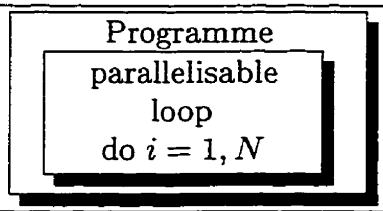


Chart 2.2 The parallelisable loop in Chart 2.1 may be turned into a subroutine of the main programme.

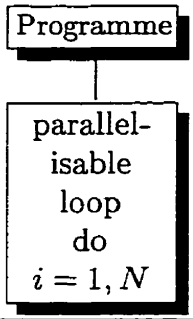
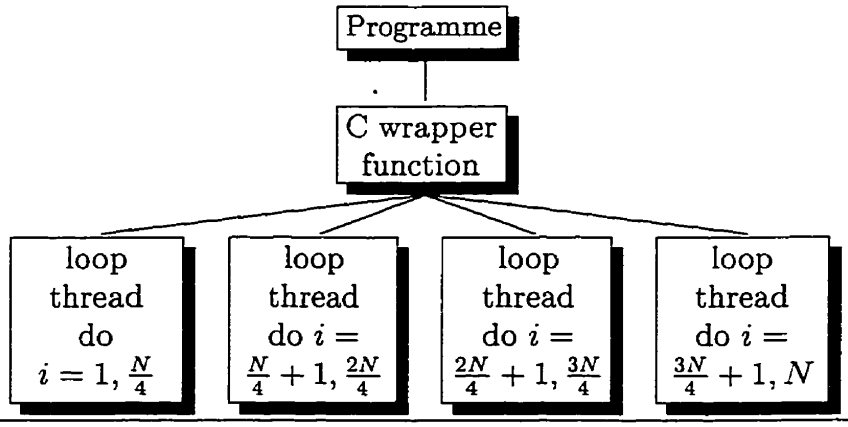


Chart 2.3 The loop may be called as threads by a wrapper function. Each thread loops through a different set of limits. Together, the threads loop through the entire range.



programme fragment, isolating the loop would lead to a code fragment much like Prog. 2.2 which could be called from the main programme as in Prog. 2.3.

```
      PROGRAM main
      INTEGER N,A(100),B(100),C(100)
      N=100
      .
      .
      .
      DO i=1,N
       C(i)=A(i)+B(i)
      ENDDO
      .
      .
      .
      END
```

Programme 2.1: A code fragment which contains a parallelisable loop. Note the absence of any dependence on the ordering of the loop index, `i`.

```
      SUBROUTINE loop(start,finish,A,B,C)
      INTEGER start,finish,A(100),B(100),C(100)
      DO i=start,finish
       C(i)=A(i)+B(i)
      ENDDO
      RETURN
      END
```

Programme 2.2: A code fragment for which the parallelisable loop in Prog. 2.1 has been isolated as a subroutine.

Before calling this subroutine using threads, we must look a bit more closely at how pthreads are called in C. The calling process is performed by the function `pthread_create`. This function requires four pieces of information: 1) the function (subroutine) to be called, 2) a structure containing the information to be passed to the function, 3) a thread to associate with the specific call, and 4) a set of attributes for the thread. Specifically, the function is called by `pthread_create(`*`thread, attributes, function, structure`*`)`.

```
      PROGRAM main
      INTEGER N,A(100),B(100),C(100)
      N=100
      .
      .
      .
      CALL loop(1,N,A,B,C)
      .
      .
      .
      END
```

Programme 2.3: How the main programme would call the isolated loop, Prog. 2.2.

Let us look at a simple C example by rewriting the subroutine Prog. 2.2 in C (Prog. 2.4). The obvious complication is the use of a structure which is passed to the function as a `void` pointer. This inconvenience permits complete freedom in choosing what parameters to pass to the function. This `void` pointer, assigned to `in`, is immediately cast to the proper structure type and assigned to `p`. Now `p` contains the information stored in the structure. Since `p` is still a pointer, the structure elements are accessed using the `pointer->element` notation.

```
typedef struct _Parameters{
 int start,finish,*A,*B,*C;
} Parameters;

void* loop(void *in) {
 Parameters *p = (Parameters *) in; //cast 'in'
 int i;
 for(i=p->start-1;i<p->finish;i++) {
  p->C[i]=p->A[i]+p->B[i];
 }
}
```

Programme 2.4: A C version of the FORTRAN subroutine, Prog. 2.2 that is suitable for calling by `pthread_create`.

The function can then be called as a thread by `pthread_create(` ***thread, attributes,*** `loop,` ***structure***`)`. A more complete implementation of this is given in Prog. 2.5.

```
#include <pthread.h>
void* loop_threads_(int *A, int *B, int *C) {
 int i;
 int N=100;
 Parameters p[4];

 /* declare the threads and the attributes */
 pthread_t thread[4];
 pthread_attr_t attributes;

 /* Set thread attributes */
 pthread_attr_init(&attributes);
 pthread_attr_setscope(&attributes,PTHREAD_SCOPE_SYSTEM);

 /* set up the parameters to pass to each threaded function */
 for(i=0;i<4;i++) {
  p[i]->start=i*N/4;
  p[i]->finish=(i+1)*N/4-1;
  p[i]->A=A;
  p[i]->B=B;
  p[i]->C=C;
 }

 /* start the threads */
 for(i=0;i<4;i++) {
  pthread_create( &(thread[i]), &attributes,
                  loop, (void *) &(p[i])     );
 }
}
```

Programme 2.5: The C wrapper function to call Prog. 2.4 in threads.

We would also like to synchronise the rest of the programme with the threads. What is needed is a place in the code beyond which the code cannot continue until all the threads are terminated. Often, this is immediately after the point where they have been called. This operation is facilitated by the pthread_join(*thread, output pointer*) function. The output pointer is similar to the void pointer used to pass the structure containing the function data to the function. However, it contains a

pointer to the structure (or data type) containing the output of the function. In the case here, there is no output, so the NULL pointer will suffice. The implementation of pthread_join in Prog. 2.5 is given by Prog. 2.6.

```
#include <pthread.h>
void* loop_threads_(int *A, int *B, int *C) {
 .
 .
 .
 /* start the threads */
 for(i=0;i<4;i++) {
  pthread_create( &(thread[i]), &attributes,
                  loop, (void *) &(p[i])     );
 }

 /* collect the threads */
 for(i=0;i<4;i++) {
  pthread_join( thread[i],NULL);
 }
}
```

Programme 2.6: The same as Prog. 2.5 but including the loop to synchronise (collect) the threads with the main programme.

This wrapper function may be called from the original FORTRAN main programme. Note how the function was named: `void* loop_threads_(int *A, int *B, int *C)`. It returns a `void` pointer, which is what FORTRAN expects. It has a trailing underscore, _, in its name, which is the default for FORTRAN objects. FORTRAN compilers often automatically append these trailing underscores. Some append a pair (g77, for example). Some append none (Cray, for example). The function takes pointers as arguments. FORTRAN passes variables by reference, so a pointer is to be expected from a FORTRAN call. Given this, the original programme (Prog. 2.3) can be easily modified to call the C subroutine (Prog. 2.5) as done in Prog. 2.7.

It would not be convenient to have to rewrite in C the FORTRAN subroutine

```
      PROGRAM main
      INTEGER N,A(100),B(100),C(100)
      N=100
      .
      .
      .
      CALL loop_threads(A,B,C)
      .
      .
      .
      END
```

Programme 2.7: How the main programme would call the C wrapper function, Prog. 2.5.

which we wish to call in a thread. In the example here, it is trivial. However, this is not always likely to be the case. So our attention must turn to the method to call the FORTRAN subroutine in a thread.

In the C vernacular, a subroutine in FORTRAN returns a `void` and its arguments are passed by reference. To call our example subroutine, Prog. 2.2, from a C code, the form given in Prog. 2.8 could be used.

```
.
.
.
int start,finish,A[100],B[100],C[100];
void loop_();
.
.
.
loop_(&start,&finish,A,B,C);
.
.
.
```

Programme 2.8: A C code fragment which calls the FORTRAN subroutine Prog. 2.2.

There is a complication when calling `loop_` as a thread, however. The thread-creation routine, `pthread_create`, expects to pass a structure to the function. FOR-

TRAN has no support for C structures. In order to circumnavigate this little problem, there must be a `void (* void)` function which can be called by `pthreads_create` which then calls `loop_` with its arguments supplied by the structure passed via a `void *`. Such a calling function is given by Prog. 2.9.

```
void loop_();

void* loop_caller(void *in) {
 /* cast 'in' to be a Parameters structure */
 Parameters *p = (Parameters *) in;
 loop_( &(p->start), &(p->finish), p->A, p->B, p->C );
 return in;
}
```

Programme 2.9: Calling function to call the FORTRAN subroutine `loop_` from a `void* (void *)` function friendly to `pthread_create`.

Now all the parts are in place to have the main FORTRAN programme call the C wrapper function which, in turn, calls the FORTRAN loop subroutine in parallel via a calling function. The main programme calls the wrapper function in the same form already seen (Prog. 2.7). The threading is done as in Prog. 2.5 by `loop_threads_` but modified to use the calling function `loop_caller` (Prog. 2.9) to give Prog. 2.10.

There is one little trick that we can use to make the coding of the wrapper function a bit easier. Recall that FORTRAN uses call by reference. That is, a pointer to a variable is passed to a subroutine instead of a copy of the variable. In our example, the addresses of the arrays `A`, `B`, and `C` are passed to the function `loop_threads` as pointers which are stored in integer pointers with the same names. The structure elements for these pointers, again of the same name, are then assigned the values of these pointers (*i.e.*, they are assigned the memory addresses of the first element of the arrays). These structure elements are then used to call the FORTRAN subroutine in which the memory addresses contained in `p->A`, `p->B`, and `p->C` are typed as integers and dimensioned as arrays.

```
#include <pthread.h>

void* loop_caller();

void* loop_threads_(int *A, int *B, int *C) {
 int i;
 int N=100;
 Parameters p[4];

 /* declare the threads and the attributes */
 pthread_t thread[4];
 pthread_attr_t attributes;

 /* Set thread attributes */
 pthread_attr_init(&attributes);
 pthread_attr_setscope(&attributes,PTHREAD_SCOPE_SYSTEM);

 /* set up the parameters to pass to each threaded function */
 for(i=0;i<4;i++) {
  p[i]->start=i*N/4+1;
  p[i]->finish=(i+1)*N/4;
  p[i]->A=A;
  p[i]->B=B;
  p[i]->C=C;
 }

 /* start the threads */
 for(i=0;i<4;i++) {
  pthread_create( &(thread[i]), &attributes,
                  loop_caller, (void *) &(p[i]) );
 }

 /* collect the threads */
 for(i=0;i<4;i++) {
  pthread_join( thread[i],NULL);
 }
}
```

Programme 2.10: The C wrapper function to start the FORTRAN subroutine `loop_` which is called by `loop_caller`.

Note that through all of this, the C wrapper function never itself uses the values of the pointers. It only uses the pointers as containers to store memory addresses which it then passes to the subroutine, `loop_`. It never knows, for example, that they are pointers to the first element of an array. The C wrapper function does not even really need to know that they are integers, since they become typed in the subroutine. For this reason, we could have just as have easily put the memory addresses into float pointers or even character pointers. However, the most generic pointer is a void pointer. So we could have written the declaration of `loop_threads` as `void* loop_threads_(void *A, void *B, void *C)`. The same would hold true for any variable passed to `loop_threads` that was not needed in the function itself. *Any variable passed to the C wrapper function which is not used in the wrapper function other than to pass to the FORTRAN subroutine may be stored in a void pointer (`void *`).* This is handy to remember and use, since it means the programmer does not need to worry about keeping subroutine arguments type-safe while setting up threads. Of course, we would have to change the structure `Parameters` to reflect these changes, since the C compiler will wish to maintain types while in C code. Such a modified structure is given in Prog. 2.11.

```
typedef struct _Parameters{
 int start,finish;
 void *A,*B,*C;
} Parameters;
```

Programme 2.11: A modified structure to take advantage of the type ambiguity of FORTRAN subroutine calls.

2.5 Dealing With Race Conditions

Consider Prog. 2.12. If the subroutine `loop` was called in parallel, there would be a definite race condition concerning the variable `sum`. As each thread attempted to update `sum`, collisions would occur leaving `sum` missing some updates. Two methods to avoid a race condition in such a situation are available. One involves special variables,

mutex locks, which control access to the variable, sum. The other has the threading wrapper function do a bit of work to sum together the separate contributions of each thread.

```
      PROGRAM main
      INTEGER A(100),B(100),C(100),sum
      .
      .
      .
      sum=0
      CALL loop(1,100,A,B,C,sum)
      .
      .
      .
      END

      SUBROUTINE loop(start,finish,A,B,C,sum)
      INTEGER start,finish,A(100),B(100),C(100),sum
      DO i=start,finish
       C(i)=A(i)+B(i)
       sum=sum+C(i)
      ENDDO
      RETURN
      END
```

Programme 2.12: A FORTRAN programme and subroutine that, if parallelised in the manner shown in Sec. 2.4 would lead to a race condition.

2.5.1 Mutex locks

Race conditions may be controlled by the use of special variables, mutex locks (MUTual EXclusion), which must be accessed prior to the variable for which a race condition may occur.

A mutex lock is simply a C structure (`pthread_mutex_t`) which contains enough information to know when it is in one of two states. These states are referred to as locked and unlocked.

The implementation of mutex locks is straightforward. For every variable for which there may exist a race condition, the programmer associates a lock. There is

no 'physical' connexion between the variable and the lock. The connexion is usually just in the naming. For example, the lock for the variable sum may be called sum_mutex. There is no way to probe a variable to see if there is a lock associated with it. Diligent use of these locks everywhere in the thread where the race variable is found is the only association. The lock is set in the locked and unlocked states with the use of the functions pthread_mutex_lock(pthread_mutex_t) and pthread_mutex_unlock(pthread_mutex_t). Once a mutex variable has been locked by a thread, all other attempts to lock that variable by other threads will cause those threads to stall until the lock is released. One does not need to worry about other threads unlocking the mutex variable since they would have had to lock it first, provided the programmer has implemented the locks in the intended fashion.

Race conditions on locks are avoided through the library itself, or by hardware means in some cases (Sun Technical Support, priv. comm.).

In FORTRAN 77, use of mutex locks requires a special modification. Recall that FORTRAN expects an implicit trailing underscore on subroutines. A call from FORTRAN code to the pthread_mutex_lock(pthread_mutex_t) function, for example, will require an object of the name pthread_mutex_lock_ at the linking stage. We must supply it with one, as is done in Prog. 2.13. A similar function may be written to deal with pthread_mutex_unlock. It may be possible, and more elegant, to simply alias the pointer to the function to a new variable named the same but with the trailing underscore.

```
#include <pthread.h>
void pthread_mutex_lock_(pthread_mutex_t *lock) {
 pthread_mutex_lock(lock);
}
```

Programme 2.13: A FORTRAN-compatible pthread_mutex_lock function.

Since it is simply a pointer that we need to pass to pthread_mutex_lock_, that

FORTRAN does not support the data type pthread_mutex_t is unimportant. The mutex variables are initialised in the C wrapper function and passed, by reference, to the FORTRAN subroutine in which the pointers can be assigned any data type. Only the address of each of the variables is passed to pthread_mutex_lock_, which is all it needs. However, any attempt to assign a value to the mutex variable within the FORTRAN subroutine will surely lead to a memory violation.

The full implementation of mutex locks in a threaded instantiation of Prog. 2.12 is given by the FORTRAN code of Prog. 2.14 and the C code of Prog. 2.15 and Prog. 2.16.

```
      PROGRAM main
      INTEGER A(100),B(100),C(100),sum
      .
      .
      .
      sum=0
      CALL loop_threads(A,B,C,sum)
      .
      .
      .
      END

      SUBROUTINE loop(start,finish,A,B,C,sum,sum_mutex)
      INTEGER start,finish,A(100),B(100),C(100),sum,sum_mutex
      DO i=start,finish
       C(i)=A(i)+B(i)
       pthread_mutex_lock(sum_mutex)
       sum=sum+C(i)
       pthread_mutex_unlock(sum_mutex)
      ENDDO
      RETURN
      END
```

Programme 2.14: The FORTRAN code required for the implementation of mutex locks in the example given in Prog. 2.12. The C code for the wrapper function loop_threads is given in Prog. 2.16 while the intermediate functions for calling pthread_mutex_unlock and pthread_mutex_unlock are given in Prog. 2.15.

```
#include <pthread.h>

/* Functions to allow the lock and unlock functions */
/*  to be called from FORTRAN */
void pthread_mutex_lock_(pthread_mutex_t *lock) {
 pthread_mutex_lock(lock);
}

void pthread_mutex_unlock_(pthread_mutex_t *lock) {
 pthread_mutex_unlock(lock);
}

/* The structure to be passed to the calling function */
typedef struct _Parameters{
 int start,finish;
 void *A,*B,*C,*sum;
 pthread_mutex_t *sum_mutex;
} Parameters;

/* The calling function */
void* loop_caller(void * in) {
 /* cast 'in' to be a Parameters structure */
 Parameters *p = (Parameters *) in;
 loop_( &(p->start), &(p->finish),
        p->A, p->B, p->C, p->sum,
        p->sum_mutex               );
 return in;
}
```

Programme 2.15: The C code containing the auxiliary functions and definitions required for the implementation of mutex locks in the example given in Prog. 2.12. The C code for the wrapper function is given in Prog. 2.16.

```
void* loop_threads_(void *A, void *B, void *C, void *sum) {
 int i, N=100;
 Parameters p[4];

 /* Declare the summation mutex variable */
 pthread_mutex_t sum_mutex;

 /* Declare the threads and the attributes */
 pthread_t thread[4];
 pthread_attr_t attributes;

 /* Initialise the mutex variable */
 pthread_mutex_init(&sum_mutex,NULL);

 /* Set thread attributes */
 pthread_attr_init(&attributes);
 pthread_attr_setscope(&attributes,PTHREAD_SCOPE_SYSTEM);

 /* set up the parameters to pass to each threaded function */
 for(i=0;i<4;i++) {
  p[i]->start     = i*N/4+1;
  p[i]->finish    = (i+1)*N/4;
  p[i]->A         = A;
  p[i]->B         = B;
  p[i]->C         = C;
  p[i]->sum       = sum;
  p[i]->sum_mutex = &sum_mutex;
 }

 /* start the threads */
 for(i=0;i<4;i++) {
  pthread_create( &(thread[i]), &attributes,
                  loop_caller, (void *) &(p[i]) );
 }

 /* collect the threads */
 for(i=0;i<4;i++) {
  pthread_join( thread[i],NULL);
 }
}
```

Programme 2.16: The C code for the wrapper function required for the implementation of mutex locks in the example given in Prog. 2.12. The FORTRAN code which calls this function is given in Prog. 2.14. Auxilliary C functions and definitions are given in Prog. 2.15.

The use of mutex locks is straightforward in its implementation. However it has a serious drawback in that it stalls threads which are waiting for a mutex variable to become free. This can become quite serious a penalty and largely negate the advantages of threads. This is particularly true of the example given, in which updating a race variable constitutes approximately half of the processing involved in the loop.

2.5.2 Eliminating race conditions by eliminating the race

An alternative to the use of mutex locks and a more elegant approach is to avoid the race condition entirely. Updates of the sort in our example are easily modified to avoid the race condition *without modification to either the main FORTRAN programme or the FORTRAN loop subroutine.* This is accomplished by passing to each instantiation of the subroutine a unique address of a variable of the same type. After the threads are finished, the wrapper function can then sum up the contributions of each thread to the original value passed to the wrapper function. The only modifications required to our example programme are in the wrapper function, `loop_threads_`, the parameter structure, `Parameters`, and the calling function `loop_caller`. Prog. 2.17 contains the necessary modifications. In the definition of `Parameters`, we see that the structure no longer contains a pointer to store the memory address of the variable `sum` but instead contains a `float` variable of the same name. This slight difference establishes the unique memory address of each thread's `sum`. Since we need to pass the address of this structure element (as it is no longer a pointer), the calling function, `loop_caller` must also be modified. The pointer was passed via `p->sum`. The memory address is instead passed via `&(p->sum)`. Summation and addition to the original variable `sum` passed to the wrapper function is done after the threads have terminated and have been collected.

Since there is no stalling in the execution of the threads while waiting for the locks to become free, the only execution penalty is within the wrapper function in

```
/* The structure to be passed to the calling function */
typedef struct _Parameters{
 int start,finish;
 void *A,*B,*C;
 float sum;
} Parameters;

/* The calling function */
void* loop_caller(void * in) {
 /* cast 'in' to be a Parameters structure */
 Parameters *p = (Parameters *) in;
 loop_( &(p->start), &(p->finish), p->A, p->B, p->C, &(p->sum) );
 return in;
}

void *loop_threads_(void *A,void *B,void *C,float *sum) {
 int i, N=100;
 Parameters p[4];

 /* Declare the threads and the attributes */
 pthread_t thread[4];
 pthread_attr_t attributes;

 /* Set the thread attributes */
 pthread_attr_init(&attributes);
 pthread_attr_setscope(&attributes,PTHREAD_SCOPE_SYSTEM);

 /* Set up the parameters to pass to each threaded function */
 for(i=0;i<4;i++) {
  p[i]->start = i*N/4+1;
  p[i]->finish= (i+1)*N/4;
  p[i]->A     = A;
  p[i]->B     = B;
  p[i]->C     = C;
  p[i]->sum   = 0; /* Initialise the summation variable */
 }
 /* Start the threads */
 for(i=0;i<4;i++) {
  pthread_create( &(thread[i]), &attributes,
                  loop_caller, (void *) &(p[i]) );
 }
 /* Collect the threads */
 for(i=0;i<4;i++) {
  pthread_join( thread[i],NULL);
 }
 /* Sum up contributions */
 for(i=0;i<4;i++) {
  *sum += *(p[i].sum);
 }
}
```

Programme 2.17: This code illustrates the use of a local array in the wrapper function to remove a race condition.

the summation of the contributions, `*sum += *(p[i].sum);`. This penalty is on the order of the number of threads, which is much less than the time taken, and saved, within the threads.

2.5.3 Avoiding race conditions by thread scheduling

It can be the case that a potential race condition may be avoided by the judicial synchronisation of the threads. Consider the subroutine in Prog. 2.18. There exists the potential for a race condition if two threads representing a continuous span in `i` are initiated at the same time. However, if the threads are separated by a span in `i` of 21 or more, there is no chance of a race condition. In our previous examples, each thread spanned 25 loops of `i`. Hence, if the threads spanning `i=1,25` and `i=51,75` were run together, no race condition could develop. Running these two threads together, collecting the threads, and then running the other pair would be the method to eliminate all race conditions without resorting to any extra computation, nor the use of mutex variables. Running threads in this fashion is an example of interlacing the threads. Prog. 2.19 gives a method to call the threads in an interlaced manner from within the wrapper function. It replaces the traditional thread creating loops seen previously.

```
      SUBROUTINE loop(start,finish,A,B,C)
      INTEGER start,finish,A(100),B(100),C(100)
      DO i=start,finish
       DO j=-10,10
        IF(i+j.gt.1.and.i+j.lt.100) then
         C(i+j)=C(i+j)+A(i)+B(i)
        ENDIF
       ENDDO
      ENDDO
      RETURN
      END
```

Programme 2.18: A subroutine for which race conditions may be avoided by interlacing the threads.

```
void* loop_threads_(void *A, void *B, void *C) {
 .
 .
 .
 /* start the threads in an interlaced manner */
 for(offset=0;offset<=1;offset++){
  for(i=0+offset;i<4;i+=2) {
   pthread_create( &(thread[i]), &attributes, loop_caller,
                   (void *) &(p[i]) );
  }
  /* collect the threads */
  for(i=0+offset;i<4;i+=2) {
   pthread_join( thread[i],NULL);
  }
 }
}
```

Programme 2.19: The code fragment of the wrapper function to call the threads in an interlaced manner.

2.6 Applying Threads to HYDRA

One may expect that a search for parallelisable loops is the first step in parallelising code. However, this may lead to time wasted spent on writing threading code for programme sections that are inconsequential to the total execution time. As well, it will inevitably overlook code that, with a bit of re-writing or clever threading, may become parallelisable.

Parallelisation of simple loops is the approach used by compilers that support automatic parallelisation. The success of such automatic parallelisation varies greatly between source codes, and generally will decrease in effectiveness with increasing source complexity. For HYDRA, automatic parallelisation provides a negligible ($< 1\%$) improvement in execution speed.

The first step in the parallelisation of a code is the determination of those parts of the code in which the bulk of the time is spent. The use of a profiling tool, such as `gprof`, which provides the amount of time spent in each subroutine makes this task

easier. Chart 2.5 shows the abbreviated call graph of HYDRA including profile data. HYDRA was run for the duration of a cosmological simulation with 2×32^3 particles. As such, it represents a 'typical' use of the code. The percentage of time spent in both the function itself and its children (descendants) is given as in Chart 2.4. The function `refforce` and its children, who calculate the forces and accelerations, clearly consume the bulk of the execution time. A more detailed call graph for `refforce` is given in Chart 2.6.

Chart 2.4 Key for the HYDRA profile trees, Charts 2.5 and 2.6.

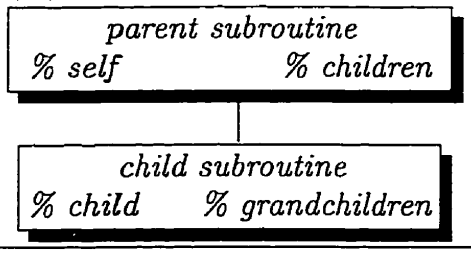


Chart 2.5 Top of the profile tree for HYDRA. See Chart 2.6 for a profile of `refforce` and its children.

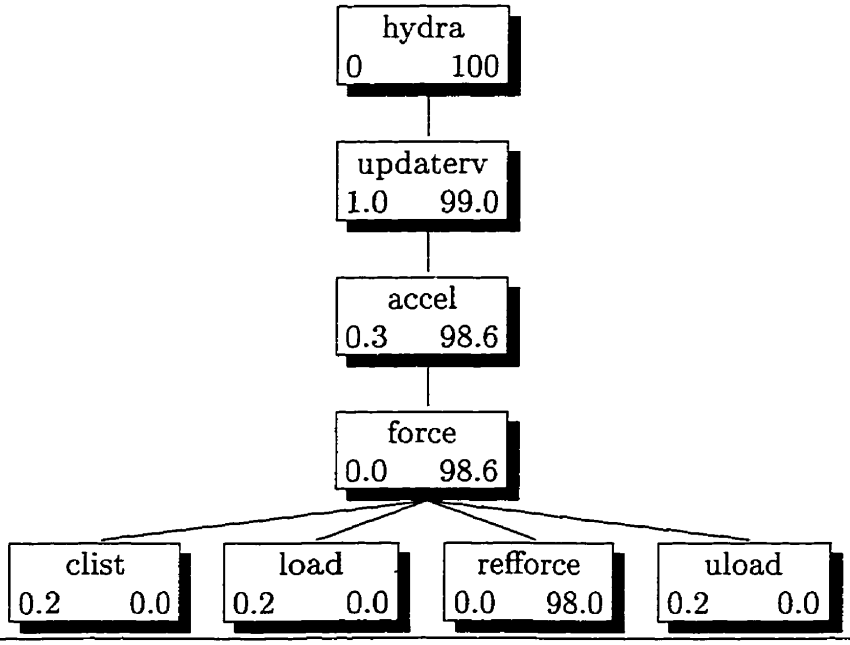
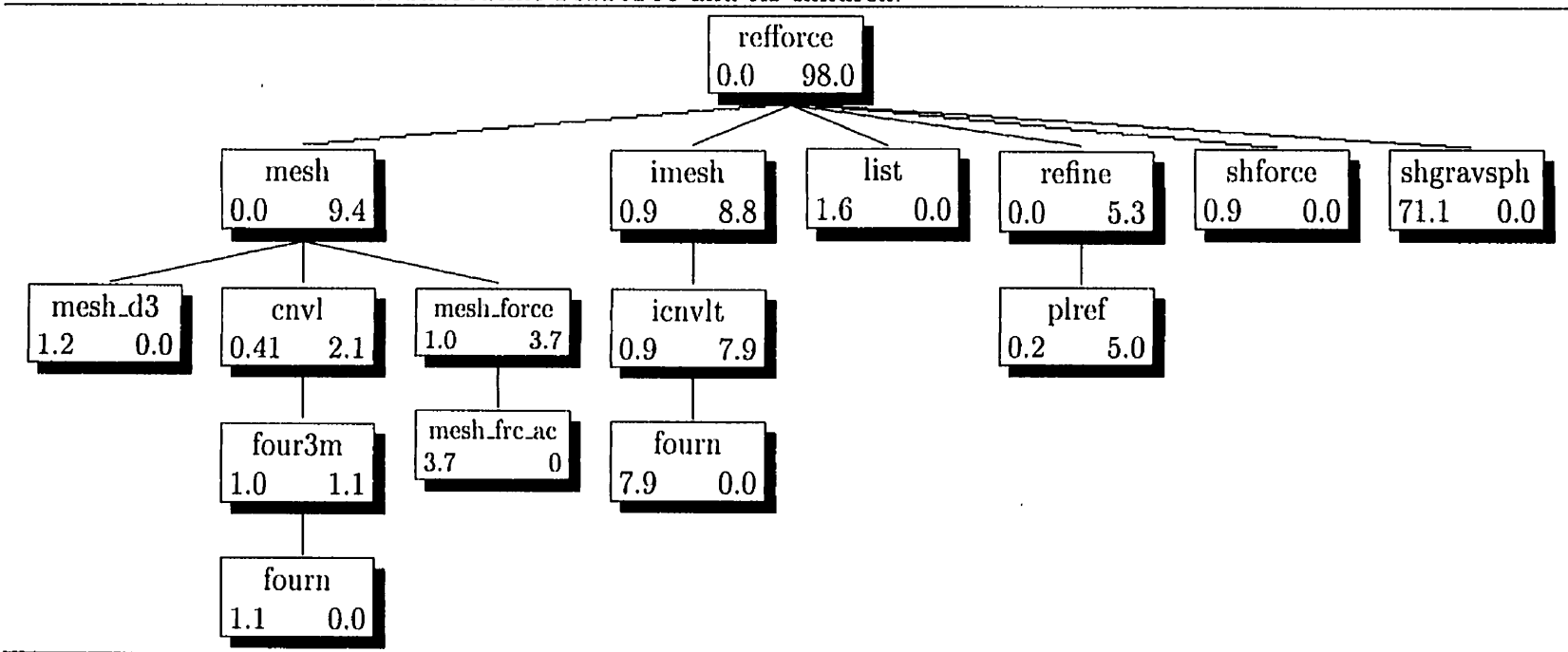


Chart 2.6 Profile tree for the subroutine `refforce` and its children.



HYDRA is an adaptive code. As a region increases in density, a refinement box is created around the region and the forces are calculated assuming the box is isolated from the larger box, except for the contribution of the global gravitational potential. In this refinement, gravitational forces may be calculated on a finer mesh than would be possible with the entire box.

It sounds reasonable to perform the force calculations on these refinements in parallel, one refinement per thread. This is a technique commonly used in the parallelisation of adaptive codes. However, this assumes that all subroutines are thread-safe. FORTRAN subroutines are not implicitly thread safe; this is compiler dependent. Hence, a different approach will be taken in which individual chunks, easy to make thread safe, will be parallelised.

2.6.1 shgravsph

The bulk of the processing time is spent within the subroutine shgravsph. Here, the short-range particle-particle gravity forces are calculated, as well as the SPH (hydrodynamic) forces. It consumes 71.1% of the processing time in the test run. Clearly, this is the bottleneck and the place to start.

Briefly, shgravsph takes as input a list containing the particles in each of the grid boxes. Each particle in a box can feel forces from every other particle in the box, and in every other particle in the surrounding boxes. It can also exert forces on these other particles. The subroutine loops over the particles in a box, and for each of these particles, loops over the particle in the box and surrounding boxes, accumulating and contributing forces. The natural loop to parallelise is the loop over each box, or at least the loop over each slab of boxes in, say, the Z direction.

Since there is overlap between boxes, adjacent boxes should not be processed in separate threads at the same time. Interlacing of the threads in the form seen in

Sec. 2.5.3 is required. This is done by having the even number grid slabs (enumerated by `ibz`) executed in one set of threads followed by the set of odd number grid slabs.

Protection of some diagnostic variables which are summed through all the boxes is also required. For this, intermediate summations are performed, akin to the method given in Sec. 2.5.2.

2.6.2 `mesh`

The subroutine `mesh`, called from `refforce`, constitutes the second most significant bottleneck. The subroutine calculates the gravitational forces on the particles estimated from their interpolation on to a grid (not the same grid seen in `shgravsph`). The first step in the procedure is the assignment of the particle masses to the grid, or mesh, to form a density field. This is performed by a subroutine extracted from the original `mesh.F` named `mesh_d3`. After this, the density field is convolved with the Green's function for the gravitational potential using a call to the routine `cnvl` to produce a gravitational potential mesh. Finally, the forces due to the potential mesh are calculated for each particle with each particle's acceleration updated accordingly. This final step occurs in the subroutine `mesh_force`, again extracted from the original `mesh.F` file.

Clearly, there is no simple loop to parallelise. Each of these sections must be examined and parallelised individually.

For the first section, `mesh_d3`, the approach to parallelisation is straightforward. The loop over the N particles is broken up into threads each containing $N/N_{threads}$ particles. There is a major caveat, however. If separate threads contain objects near the same mesh node, there exists a potential for a race condition. Since there is potentially little correlation between particle positions and particle number, interlacing will not help remove this race condition. The use of separate density meshes for each

thread would be expensive memory-wise and the final summation would be CPU intensive. Locks would be required for each node, if they were to be used. There would also be a significant amount of overhead in locking and unlocking 8 times for each particle, irrespective of collisions. For this reason, the race condition is allowed to proceed. Tests indicate that this race condition leads to a small, but non-negligible error of approximately $1 \times 10^{-8}\%$. The option exists at compile time to choose not to parallelise this section.

The convolution section, `cnvl` is not directly parallelisable. The bulk of its execution time is spent in `four3m` doing three dimensional Fourier transforms. This subroutine is parallelisable. It must be done in two sets of threads. Each of these loop over layers of the mesh without any complication.

The final section, the calculation of the gravitational forces at the mesh nodes and their interpolation to each particle is a two step process. Both are parallelisable, however the bulk of the time is taken during interpolation. This loop was parallelised relatively easily by breaking up the loop over all particles into separate threads for subsets of the particles.

2.6.3 `imesh`

This subroutine provides the same function as `mesh`, seen above, but performs the task assuming non-periodic boundary conditions. It is used during refinements to calculate the large-scale gravitational forces within the refinement box. Hence, it operates on a small fraction of the total number of particles. Because of this, the loops seen in `mesh` over the total number of particles are not nearly as significant to the total execution time. What remains significant is the time spent in the convolution. This is set by the mesh resolution which generally remains constant relative to the box size. That is, the number of nodes in the mesh is fairly constant.

The subroutine `icnvlt` performs the convolution for the isolated mesh. It can be broken into three separate parallelisable loops. All three loop over mesh layers. All are standard in their implementation. The second loop requires the creation of a preliminary data table which is normally calculated iteratively within the loops. There is minimal overhead in this operation.

2.6.4 `plref`

Called from `refine`, `plref` controls the placement of the refinements. The refinements are calculated in four passes, of which two are easily parallelisable. Fortunately, this pair consumes the most processing time.

Because `plref` calls further subroutines and functions, care was taken to ensure that these other codes were thread-safe.

2.6.5 `updaterv`

The updating of each particle's position and velocity are performed by the subroutine `updaterv`. HYDRA uses a Predictor-Evaluation-Corrector (PEC) update algorithm. For this reason, the particle attributes are updated twice in this routine. Updating values requires a trivial loop over all the particles. It is equally trivial to parallelise. Despite the knowledge that they do not consume a great deal of CPU time, these two loops were threaded owing to the simplicity of the operation.

One of the particle attributes updated in the second loop is the thermal energy. This requires a call to the cooling function which calls other subroutines. These functions have not been yet made thread-safe, so a mutex variable was used to lock this function so that only one thread can access the function at a time. A lock was also used to protect the seldom-called function `lostp` which is invoked when a particle in an isolated top-level box (not to be confused with an isolated refinement) spills out of the box boundary.

2.7 Performance

The effectiveness of the parallelisation described previously can be measured by the apparent fraction of the code running in parallel, f_p. This can be calculated using the time required by a code running in serial, t_s, the time required to run the same code by a parallelised version of the code, t_p, and the number of CPUs, N. The time t_p relates to t_s by

$$t_p = t_s \left(\frac{f_p}{N} + f_s \right) \tag{2.2}$$

where f_s is the unavoidable fraction of the code that does not run in parallel. Since $f_p + f_s = 1$,

$$t_p = t_s \left(1 + f_p \left(\frac{1}{N} - 1 \right) \right) \tag{2.3}$$

from which it falls that,

$$f_p = \frac{1 - \frac{t_p}{t_s}}{1 - \frac{1}{N}} \tag{2.4}$$

Through a cosmological simulation, this value changes as the bulk of the processing shifts among the subroutines. Table 2.1 demonstrates the changes to f_p as a simulation evolves from a uniform distribution of particles at $a \approx 0$ to a highly clustered state with many refinements at $a = 1$. Clearly, as the number of refinements increases, the effectiveness of the modifications to the code described here decreases. The shift in weight among the various subroutines changes as outlined in Fig. 2.1. This figure verifies the shift from subroutines that have been parallelised to those that have not. However, it does not explain the extent of the drop in performance. The remaining performance drop is attributable to inefficient balancing of the load among the threads. This is a problem for `shgravsph` in which the particles sent to threads are grouped by position among a series of slabs. Under the extreme clustering exhibited

	Measured			Corrected			
a	t_p	t_s	t_{corr}	t_p	t_s	$f_p(\%)$	$n_{refinements}$
≈ 0	3:30.62	5:48.23	37.40	173.22	310.83	88.5	0
0.2	9:32.58	14:53.60	49.72	513.86	844.88	78.4	44
0.4	19:59.70	28:44.40	69.04	1130.66	1655.36	63.4	149
0.6	27:17.33	37:39.17	76.81	1560.52	2182.36	57.0	192
0.8	32:41.86	45:37.55	86.65	1875.21	2650.90	58.5	205
1.0	38:49.50	57:26.14	87.26	2242.24	3358.88	66.5	198

Table 2.1: The performance of the parallelised code as it changes with the cosmological expansion factor for the simulation, a. The execution times of the code (in seconds) when run with parallelisations (t_p) and without (t_s) are corrected for overhead in the tests due to data loading (t_{corr}). The times are for 10 time steps of a cosmological simulation with 2×64^3 particles. The effective fraction of the code running in parallel, f_p is given as a percentage of the code. This can be compared with the number of refinements per time step, $n_{refinements}$.

in the latter stages of the simulation, this leads to an imbalance in the number of particles dealt each thread.

The performance of the enhancements decreases with the size of the simulation (Table 2.2). This is contrary to the Modified Amdahl's Law. This states that the performance gain from parallelisation is greater than or equal to that which Amdahl's Law would imply since parallelisation usually permits larger problems to be run which, in general, shifts more of the processing to those parts of the code which are parallelised. In the case of cosmological simulations with this version of HYDRA, a larger problem means higher resolution and, consequently, more refinements. It is the unparallelisability (in the present form), of the refinement list creation that is again the bottleneck.

2.8 Portability

The use of the pthreads library promises good portability. However, FORTRAN itself may be the limiting factor. Since many FORTRAN libraries are created without

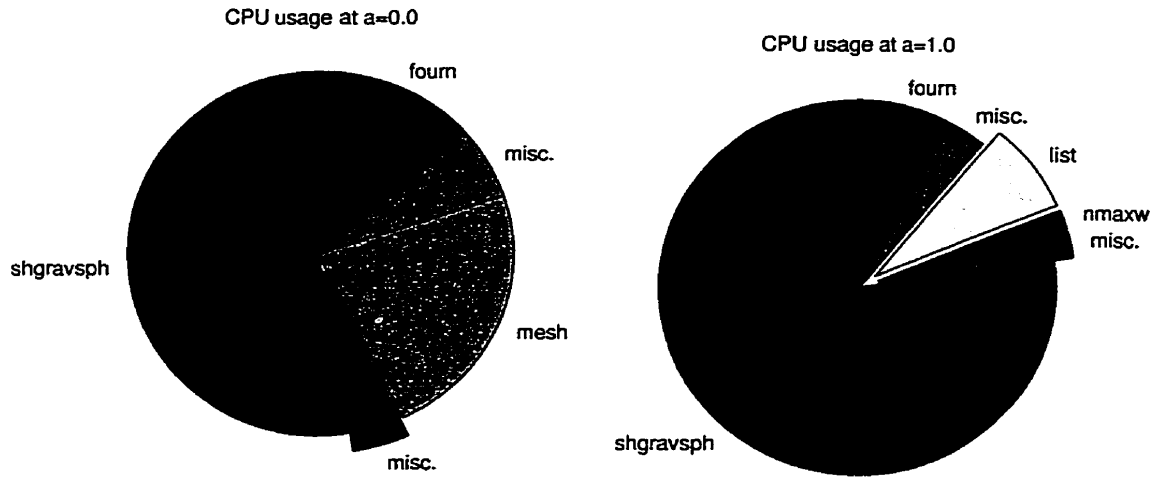


Figure 2.1: The dominant subroutines at early and late times. Each plot shows the fraction of CPU time consumed by the dominant subroutines. The left-hand plot is for early times ($a \approx 0$) and the right-hand plot corresponds to the end of the simulation time ($a = 1$). Segments of the code which are not parallelised are separated from the chart.

	f_p	
a	64^3	32^3
≈ 0	76	94
0.2	72	91
0.4	47	68
0.6	44	50
0.8	46	45
1.0	53	44

Table 2.2: The scaling of the parallelised code. Given is the effective fraction parallelised, f_p, for simulations with 64^3 and 32^3 gas particles at different points in the evolution, denoted by the expansion factor, a. These tests were run on a dual-processor i686 Linux box. The factors were not corrected for i/o overhead.

parallelisation in mind, the libraries themselves may not be thread-safe. Limited testing of the portability of this code beyond a Solaris environment was done (Linux and DEC boxes). In one instance (a dual-processor i686 running Linux) the code compiled and ran without any modification. In the case of the DEC box, results were not so satisfying. It is essential that a thread-safe FORTRAN library be used. Even the label, thread-safe, can be misleading as it was found that the `mod` function of one older "thread-safe" library was found to be corruptible. Debugging threaded code is not always easy, so these are serious concerns. However, thread-safe libraries are becoming increasing prevalent, often, as in the case of Solaris, to be used with propriety threading libraries.

Chapter 3

SPH ALGORITHM TESTS

As part of a series of tests of the N-body hydrodynamical method, smoothed particle hydrodynamics (SPH), the cooling behaviour near cold dense clumps and the drag these clumps encounter in hot halos were examined. This series of tests explored a selection of situations relevant to cosmological simulations and known to cause SPH problems (Thacker et al. 1998). The tests were performed using a variety of implementations of SPH. Here, a brief description of the implementations will be given, followed by an examination of the difficulties encountered by the SPH algorithm in the presence of steep density gradients. Finally, results for the cooling and drag tests will be presented for the various implementations.

3.0.1 Smoothed Particle Hydrodynamics

The particle method for modelling fluid flow introduced simultaneously by Gingold and Monaghan (1977) and Lucy (1977) is well suited for use in N-body simulation codes which model the evolution of the mass distribution using Lagrangian particles. In essence, a property at any position is calculated to be the average of that property of the surrounding particles. The average is weighted with distance by a kernel that is a smoothly varying function of distance. Hence the technique is labelled 'smoothed

particle hydrodynamics'.

The average of any scalar quantity, $A(\mathbf{r})$, is given simply as

$$\langle A(\mathbf{r})\rangle = \int d\,\mathbf{r}'\, A(\mathbf{r}')W(\mathbf{r}-\mathbf{r}', h), \tag{3.1}$$

where $W(\mathbf{r}, h)$ is the kernel function and h is the smoothing length. The kernel function is generally, but not necessarily, spherically symmetric. As well, it is normalised such that

$$\int d\,\mathbf{r}'\, W(\mathbf{r}-\mathbf{r}', h) = 1. \tag{3.2}$$

The smoothing length, h, is generally related to the kernel in that W is compact over (goes to zero beyond) $2h$. Hence, $W = W(x)$, with $x \equiv \mid \mathbf{r} \mid /h$ and $0 \leq x \leq 2$. For a discrete system of particles, this average becomes

$$\langle A(\mathbf{r})\rangle = \sum_{j}^{N} \frac{A(\mathbf{r}_j)}{n_j} W(\mathbf{r}_i - \mathbf{r}_j, h), \tag{3.3}$$

where n_j is the number density around the j^{th} particle. The number density is often seen replaced with ρ_j/m_j where m_j and ρ_j are the mass and mass density, respectively. The number density for each particle, n_i, can be calculated similarly as

$$\langle n\rangle_i = \sum_{j}^{N} W(\mathbf{r}_i - \mathbf{r}_j, h). \tag{3.4}$$

This is derived from the discrete form of the normalisation equation, Eq. 3.2.

In practice, the number density is the only quantity that need be found by the SPH summation; mass is invariant while temperature and velocity are set by balancing the force and energy equations. The pressure of a gas particle is simply a function of density and temperature. However, the gradient of the pressure field is required to calculate the hydrodynamic forces. The advantage of the SPH formalism is that

the gradient of the scalar field can be calculated, according to Monte Carlo theory, by simply using the gradient of the kernel function in the manner,

$$\nabla\langle A\rangle_i = \sum_j^N \frac{A(\mathbf{r}_j)}{n_j}\nabla W(\mathbf{r}_i - \mathbf{r}_j, h). \tag{3.5}$$

Clearly it is required that $\nabla W(\mathbf{r}_i - \mathbf{r}_j, h) = -\nabla W(\mathbf{r}_j - \mathbf{r}_i, h)$.

In this formalism, the smoothing length, h, has been taken to be a constant. However, in cosmological simulations in which there is a great deal of clustering, this would be inappropriate since it would establish a minimum resolution far larger than the resolution permitted by the increased number density in the dense regions. The smoothing length should decrease as local number density increases to maintain an appropriate spatial resolution. For h to scale the resolution with the number density, h_i can be taken to be a distance which encompasses some number of neighbouring particles, N_{SPH}. Indeed, the SPH approximation to hydrodynamics converges to the exact solution as $N \to \infty$ and $N_{SPH} \to \infty$, albeit at a slower rate (Lombardi, Sills, and Shapiro 1998).

This modification leads to the question of which particle's smoothing length should be used in the SPH summations. There are two interpretations of SPH: the 'gather' and the 'scatter'. In the 'gather' interpretation, a local estimate of a property is made using that value of the local particles *sampled out to some fixed distance* and weighted by their distance from the location relative to the fixed distance. Hence, the smoothing length in the summation would be h_i. In the 'scatter' interpretation, the estimate is made using the sum of the *contributions to that position* from all the particles for which their smoothing length extends to encompass the position. Hence, h_j would be used. In practice, neither interpretation is accepted. The appropriate smoothing length is, instead, some function of the individual h's, *i.e.*, $h_{ij} = f(h_i, h_j)$ such that h_{ij} is in the range of h_i to h_j. This is referred to as the averaging of h since

the function is generally some averaging technique. This average can be arithmetic, harmonic, geometric, or some other form. It need not be symmetric in the general case (for example, to calculate densities), but if it is to be used in the calculation of force, it must be. As another option, it is the kernels that are averaged, such that $W(\mathbf{r}_i - \mathbf{r}_j, h)$ is replaced with $\left(W(\mathbf{r}_i - \mathbf{r}_j, h_i) + W(\mathbf{r}_i - \mathbf{r}_j, h_j)\right)/2$.

The kernel itself is generally Gaussian in form but goes to zero, as mentioned, beyond a distance of $2h$. The details of the kernel used here can be found in Thomas and Couchman (1992).

The contributed force, F, from hydrodynamics to the acceleration on each particle comes from the pressure gradient, ∇P, via

$$F = -m\frac{\nabla P}{\rho}, \tag{3.6}$$

where m and ρ are the gas particle mass and density, respectively. This form implies that we can use Eq. 3.5. However, since $\nabla\langle A\rangle_i$ is not necessarily equal to $\nabla\langle A\rangle_j$ in Eq. 3.5, the force calculation may not be symmetric, which is, of course, unphysical. It is necessary to find an SPH approximation of the hydrodynamical force contribution such that $F_{ij} = -F_{ji}$ where F_{ij} is the force felt by particle i due to particle j. Two methods are described here which both use the identity

$$\frac{\nabla P}{\rho} = \nabla\frac{P}{\rho} + \frac{P}{\rho^2}\nabla\rho. \tag{3.7}$$

Using this and Eq. 3.5, Eq. 3.6 expands into,

$$\begin{aligned} F_i &= -m_i\left(\sum_j^N m_j\frac{P_j}{\rho_j^2}\nabla W(\mathbf{r}_i - \mathbf{r}_j, h_j) + \frac{P_i}{\rho_i^2}\sum_j^N m_j\nabla W(\mathbf{r}_i - \mathbf{r}_j, h_j)\right) \\ &= -m_i m_j\left(\sum_j^N \frac{P_j}{\rho_j^2}\nabla W(\mathbf{r}_i - \mathbf{r}_j, h_j) + \sum_j^N \frac{P_i}{\rho_i^2}\nabla W(\mathbf{r}_i - \mathbf{r}_j, h_j)\right). \end{aligned} \tag{3.8}$$

This is not yet symmetric since $\nabla W(\mathbf{r}_i - \mathbf{r}_j, h_j) \neq -\nabla W(\mathbf{r}_j - \mathbf{r}_i, h_i)$ if $h_i \neq h_j$. For a spatially varying smoothing length, Monaghan (1992) suggests taking the gradient of some average of the kernels calculated going both ways, giving,

$$F_i = -m_i m_j \sum_j^N \left(\frac{P_j}{\rho_j^2} + \frac{P_i}{\rho_i^2} \right) \nabla \bar{W}_{ij}. \tag{3.9}$$

The 'average' kernel, $\bar{W}_{ij}$, can use the kernel averaging scheme seen above or some average of the smoothing lengths for h_{ij}. That is,

$$\bar{W}_{ij} = \left[\nabla W(\mathbf{r}_i - \mathbf{r}_j, h_j) + \nabla W(\mathbf{r}_i - \mathbf{r}_j, h_i) \right] / 2 \tag{3.10}$$

or $\bar{W}_{ij} = \nabla W(\mathbf{r}_i - \mathbf{r}_j, h_{ij})$ where h_{ij} is an average of the smoothing lengths seen previously. This approach is used in Steinmetz and Müller (1993) (hereafter, SM93).

A method particularly efficient in its use of memory is used by Thomas and Couchman (1992) (hereafter, TC92). They suggest assuming that $\nabla W(\mathbf{r}_i - \mathbf{r}_j, h_j) \approx -\nabla W(\mathbf{r}_j - \mathbf{r}_i, h_i)$ which is however not correct to order ∇h. This gives

$$F_i = -m_i m_j \left(\sum_j^N \frac{P_j}{\rho_j^2} \nabla W(\mathbf{r}_i - \mathbf{r}_j, h_j) - \sum_j^N \frac{P_i}{\rho_i^2} \nabla W(\mathbf{r}_j - \mathbf{r}_i, h_i) \right) \tag{3.11}$$

which is the form implemented in the production version of HYDRA.

Early on, it was realised that a viscosity force term would be necessary to damp the flow of gas (see, for example, Lucy (1977)). This is entirely to be expected; the gas is represented by simulation particles of macroscopic size, while viscosity is due to turbulence on all scales, including microscopic. The adopted forms of the viscosity terms are essentially *ad hoc* with empirical tweaking. They are all designed to cover bulk viscosity and shock front dissipation (Monaghan 1992) as well as shear in some cases. The production version of HYDRA, as outlined in TC92, utilises an additional

term, P_{visc}, to each particle's pressure term which is activated in the presence of convergent flow,

$$P_{visc} = \begin{cases} \rho_i \left[-\alpha c_i h_i \nabla \cdot \mathbf{v}_i + \beta (h_i \nabla \cdot \mathbf{v}_i)^2 \right], & \nabla \cdot \mathbf{v}_i < 0 \\ 0, & \nabla \cdot \mathbf{v}_i \geq 0, \end{cases} \tag{3.12}$$

where c_i is the speed of sound, α and β are empirically derived coefficients with values of 1 and 2 respectively.

To create an artificial viscosity, Monaghan (1992) suggests an extra term, Π_{ij}, in the force equation (Eq. 3.6) which uses a per-particle convergence trigger. It gives for this term,

$$\Pi_{ij} = \begin{cases} \frac{-\alpha \langle c \rangle_{ij} \mu_{ij} + \beta \mu_{ij}^2}{\langle \rho \rangle_{ij}}, & \mathbf{v}_{ij} \cdot \mathbf{r}_{ij} < 0 \\ 0, & \mathbf{v}_{ij} \cdot \mathbf{r}_{ij} > 0, \end{cases} \tag{3.13}$$

where

$$\mu_{ij} = \frac{h \, \mathbf{v}_{ij} \cdot \mathbf{r}_{ij}}{r_{ij}^2 + (0.1 \langle h \rangle_{ij})^2}. \tag{3.14}$$

The $(0.1\langle h \rangle_{ij})^2$ term in the denominator of μ_{ij} prevents division by zero while remaining small enough at all number densities to not smooth the viscosity over too many particles. Here, $\langle A \rangle_{ij}$ is simply the arithmetic average of A_i and A_j.

These forms, particularly the Monaghan form, damp shear flows. This is not always wanted. For example, in disc formation this leads to a transfer of angular momentum and a damping of the rotation of the cloud. In shear flows, $\nabla \cdot \mathbf{v} = 0$ and $|\nabla \times \mathbf{v}| > 0$, whereas $\nabla \cdot \mathbf{v} < 0$ and $|\nabla \times \mathbf{v}| = 0$ in purely compressional flow. Hence, $\nabla \times \mathbf{v}$ can be used as the 'trigger' in a shear-correction factor to the viscosity term so that $\Pi_{ij} \rightarrow \Pi_{ij} \langle f \rangle_{ij}$ which was first suggested in Balsara (1995). Steinmetz (1996) gives for the factor,

$$f_i = \frac{|\langle \nabla \cdot \mathbf{v} \rangle_i|}{|\langle \nabla \cdot \mathbf{v} \rangle_i| + |\langle \nabla \times \mathbf{v} \rangle_i| + 0.0001 c_i / h_i}, \tag{3.15}$$

Version	Artificial Viscosity	symmetrisation
1	TC92	TC92
2	TC92+shear correction	TC92
3	local TC92	TC92
4	Monaghan	arithmetic h_{ij}
5	Monaghan	harmonic h_{ij}
6	Monaghan	kernel averaging
7	Monaghan+shear correction	arithmetic h_{ij}
8	Monaghan+shear correction	harmonic h_{ij}
9	Monaghan+shear correction	kernel averaging
10	Monaghan	TC92
11	Monaghan $\langle\rho\rangle_{ij}$	TC92
12	Monaghan $\langle\rho\rangle_{ij}$	TC92 + kernel av

Table 3.1: Summary of the implementations examined. The terms are discussed in the text. Adapted from Thacker et al. (1998).

again with a term in the denominator to remove the possibility of a singularity.

3.0.2 The test implementations of SPH

Two aspects of the implementation of SPH are varied in these tests: the method of symmetrising the force contribution due to the pressure gradient, and the form of the artificial viscosity. There are twelve versions of the code, each with a separate combination of variations (Table 3.1). Again, details of the coding can be found in Thacker et al. (1998) and it was Thacker who coded the implementations.

The symmetrisation methods tested were either those of TC92 or SM93. In one implementation, the TC92 symmetrisation is augmented by kernel averaging. This is similar to the symmetrisation of SM93, but the neighbour search is different. For SM93 symmetrisation, there are three variations of the symmetrisation of the kernel gradient: either arithmetic averaging of h, harmonic averaging of h, or kernel averaging.

The artificial viscosity term was varied in the tests between the $\nabla \cdot \mathbf{v}$ version of

TC92 and the $\mathbf{v}_{ij} \cdot \mathbf{r}_{ij}$ version of Monaghan. For each, there are implementations which include the shear-correction factor. One variant of the $\nabla \cdot \mathbf{v}$ viscosity was also used which traded lower shot noise for higher resolution by calculating $\nabla \cdot \mathbf{v}$ over fewer particles. The $\mathbf{v}_{ij} \cdot \mathbf{r}_{ij}$ version also had one variant which calculated $\langle \rho \rangle_{ij}$ in Eq. 3.13 by estimating it from $\rho_i(1 + (h_i/h_j)^3)/2$. In this case, the shear-correction term was not an average of the factors f_i and f_j; only f_i was used. This trades increased accuracy for computational efficiency.

3.1 SPH Near Steep Density Gradients

It is known that the SPH estimation of the density for a particle fails near a steep density gradient. The density is essentially determined by finding the radius of a sphere that encompasses some given number of particles, N_{SPH}, which is typically 20-30. That the particle contributions are weighted by a smoothly-varying kernel function is a second-order correction to the estimate. Crudely, in the presence of a varying density gradient, contributions from particles in the higher-density region exceed those from the lower-density region, leading to an over-estimate of the local density. Non-spherically symmetric kernels alleviate this problem, but are computationally expensive both in memory resources and CPU time. In this section, this phenomenon will be explored in more detail with the aim to predict those gradients that suffer this effect most strongly.

3.1.1 The density estimate in a medium with no gradient

In the presence of no density gradient, the density estimate should be without error. However, this will be true if and only if every particle 'sees' a uniform density. Whether this is true is dependant on the distribution of particles. We will look at three density distributions: uniform (regularly spaced), glass, and Poisson. The glass distribution was kindly provided by R. Thacker.

In the SPH formalism, the density at any point can be calculated by the summa-

tion of the kernel-weighted contributions of local particles. In practice, the point in space for which the density is required is the position of a particle. There are two consequences of this. The densities calculated are essentially mass weighted and, as such, if there is any degree of clumping then the mean of all the densities calculated will be greater than the volume-weighted density (total mass / total volume). The second consequence is the presence of a base self-density in every density calculation. This base self-density is simply the contribution to the density of a particle at a radius of zero from the position. This consequence is intimately related to the first in that it implies, again, that the mean of the densities calculated will always be greater than the volume-weighted density since there is always a particle very near (in this case, at) the position at which the density is being calculated.

In normal uses of SPH, these aspects of the SPH density calculation are not unwanted artifacts. Indeed, the densities required for the force calculations are the densities *at the particle positions.* They do lead to a problem here, however, where a comparison of the mean of the densities calculated to the actual mean density is required. Of particular interest is the behaviour of the SPH density calculation in a Poisson distribution of particles, which is the easiest non-regular distribution to create. For density gradients, it is difficult to use regular distributions without creating preferred directions beyond the direction of the density gradient. That is, the crystalline lattice structure can create unwanted effects. Glasses (distributions of particles akin to the distribution of atoms in glass) are difficult to produce since they require evolutionary relaxation through an N-body code. In this section we will see that this relaxation would need to be done for every different density slice, as well, to prevent regular patterns dominating at high densities. That is, we can not simply tile one distribution.

The two point correlation functions for the distributions are plotted in Fig. 3.1.

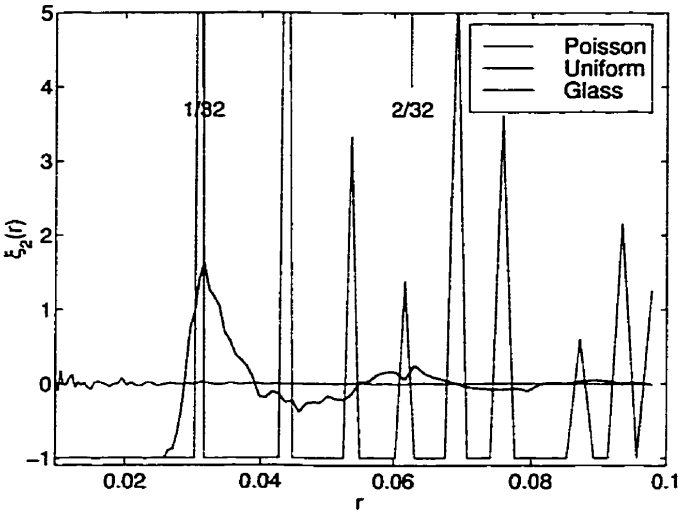


Figure 3.1: The two point correlation function for the constant-density distributions. The two point correlation function, $\xi_2(r)$ is plotted for the Poisson (red), uniform (green), and glass distributions (blue). These distributions are for volumes with 32^3 particles. The distances corresponding to 1/32 and 2/32 are marked.

This function essentially plots the excess probability of finding a particle at the distance r from any given particle compared with a constant density distribution. For a Poisson distribution, there is no correlation, which is verified by the nil excess probability at all distances, allowing for scatter from low-number statistics at close distances. For the uniform distribution laid out on a mesh, there are peaks representing the distances to the mesh nodes. There can be no particles within 1/32 for this mesh with 32^3 nodes. For this reason, there is complete anti-correlation at distances less than 1/32. The anti-correlation of the glass distribution at distances less than this is also evident, a by-product of the system adopting an amorphous pseudo-crystalline configuration during the relaxation.

The effect of the self-density contribution to the estimate of the *volume-weighted* density is illustrated in Fig. 3.2. Clearly demonstrated is the convergence as N_{SPH} increases, independent of either the presence of the self-density term or the form of the distribution. For the glass and uniform distributions, the densities converge from

either below (under-estimation of the density) or above (over-estimation), dependant on the presence of the self-density term. In contrast, for the Poisson distribution the density estimate converges immediately, albeit with large scatter, in the absence of the self-density term. In the presence of this term, it converges slowly, always over-estimating the density. The absence of correlation at all scales, as see in Fig. 3.1, explains this behaviour as a manifestation of the first of the two consequences of the self-term term described above. Though there is no correlation between particles, there is still clumping, as when two particles are (randomly) near each other. So it should be expected that there would be an over-estimation of the density. In the absence of the self-density term, the density summation over particles becomes one of summation about a randomly chosen point in space. In the SPH formalism, then, this *should* calculate the volume-weighted density. In the other two distributions, their anti-correlation at small distances implies that exclusion of the self-density term in the summation equates to preferentially sampling *voids*, which leads to the under-estimate of the mean density.

The same analysis can be done for an SPH calculation of the density that fixes the kernel radius, h, instead of the summation number, N_{SPH}. This reveals that the mean density calculated for the Poisson distribution, without using the self-density term, is indistinguishable from that of the glass distribution calculated with the self-density term (Fig. 3.3). It is the calculation with the self-density term that consistently produces the correct value for the mean density of the glass distribution.

Since the following analysis compares the mean of the calculated densities with the volume-averaged density in a series of slices with Poisson distributions, we will be using an SPH summation *without* the self-density.

3.1.2 The density estimate in a medium with a gradient

For any spherically symmetric weighting kernel, the SPH estimate of the density for a

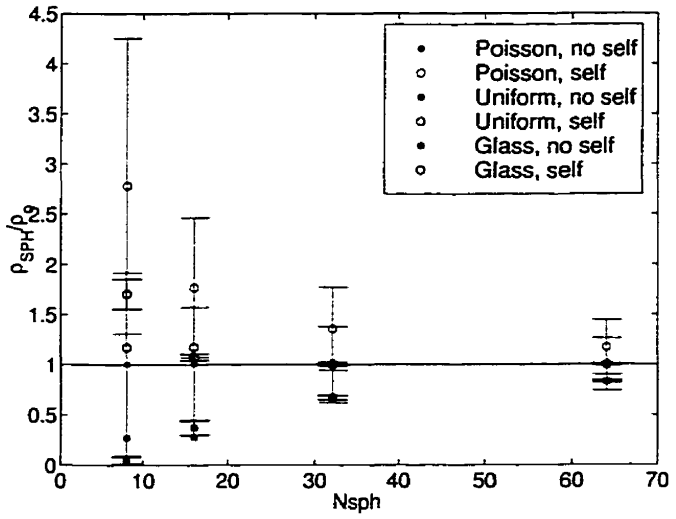


Figure 3.2: The error in the SPH density estimate *vs.* N_{SPH} for a constant density volume. Shown are the mean densities (with errors) over all particles when the value for N_{SPH} is varied. Density was calculated both including the self-density (asterisk) and without (open circles). The densities calculated in the Poisson distribution are given in red, while those calculated in the uniform and glass distributions are given in green and blue, respectively. The data for the uniform and glass distributions overlap for $N_{SPH} = 32$ and 64.

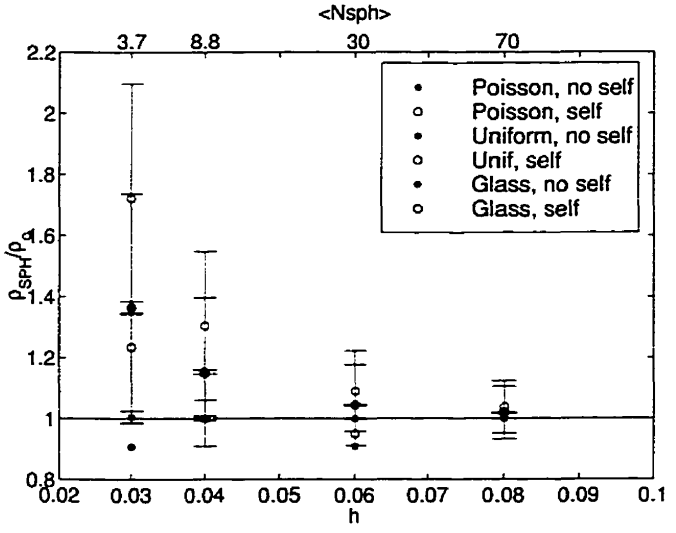


Figure 3.3: The error in the SPH density estimate *vs.* h for a constant density volume. Shown are the mean densities (with errors) over all particles when the value for h is varied. Density was calculated both including the self-density (asterisk) and without (open circles). The densities calculated in the Poisson distribution are given in red, while those calculated in the uniform and glass distributions are given in green and blue, respectively. The data for the uniform and glass distributions overlap for $h = 0.04$ and 0.08. At the top, the mean values of N_{SPH} are given.

particle in a *constant* density gradient will produce no error. This requires, of course, that the gradient be constant over the span of $x - 2h < x < x + 2h$ which implies that the gradient is resolved by the particle density.

The toy model that will aid us in our exploration will be a volume with an increasing density gradient in one direction. Specifically, the density will behave as $\rho \propto x^{-1}$. This is a similar form to the density profiles found in cosmological objects. An iso-density surface for these objects, though spherical in morphology, is locally flat. Hence this toy model is relevant to the density gradients found in cosmological objects.

The models were constructed from 100 slices, each with the same number of particles but with widths, W_i, such that $FW_{i-1} = W_i = 1/FW_{i+1}$ where $F \geq 1$ and is set here to 1.1 or 1.5. The particles were distributed randomly.

In our tests, the mass per particle is unity, giving a number density, n, equivalent to the mass density, ρ.

Error in the density estimate arises from both statistical variance as well as the systematic error due to the density gradient. The former error is reduced by using more particles in the estimate of the density. The latter is reduced by kernel averaging over a smaller region, and hence using fewer particles. The regime of dominance of these effects is illustrated in Fig. 3.4. Near the density cusp on the right, where gradients are higher, the density estimate which kernel averages over fewer particles (smaller value of N_{SPH}) displays large variance, but no systematic over-estimation of the density. In comparison, the estimates using the larger number of particles systematically over-estimate the density by a factor of 10 to 20%. Further away from the cusp, indeed over the bulk of the range, the larger values of N_{SPH} allow density estimates with smaller amounts of statistical variance and no systematic bias.

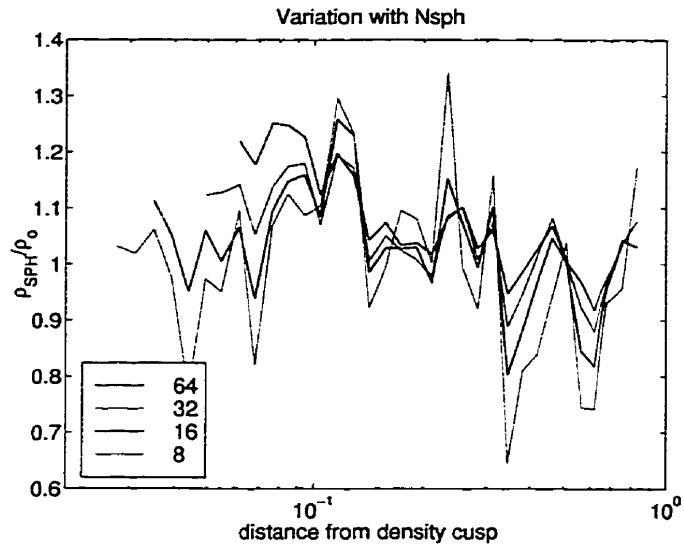


Figure 3.4: The error in the SPH density estimate as it varies with N_{SPH}. Shown are the errors in the density estimate for sets of data as they vary with distance from the density cusp. The highest density gradients are found near the density cusp. The sets of data correspond to differing values of N_{SPH}, the number of particles used for the local estimate of density.

Increasing the number density of particles increases the resolution of the simulation. This permits steeper gradients to be resolved. For the $\rho \propto x^{-1}$ profile of our tests, this behaviour is qualitatively verified in Fig. 3.5.

More useful is a parametrisation of the density gradient that is independent of the local number density of particles. Consider the parametrisation based on the change of the density over the smoothing distance, $2h$, given by $\rho(x-h)/\rho(x+h)$. As seen in Fig. 3.6, this parametrisation indicates that SPH fails (at the 20% level) when the density across a smoothing length varies by more than a factor, F_c, or 3-4 times, for the $\rho \propto x^{-1}$ profile used here.

An estimate of the radial distance from the centre of a dense clump at which SPH will begin to calculate erroneous density values can be made using F_c. This factor,

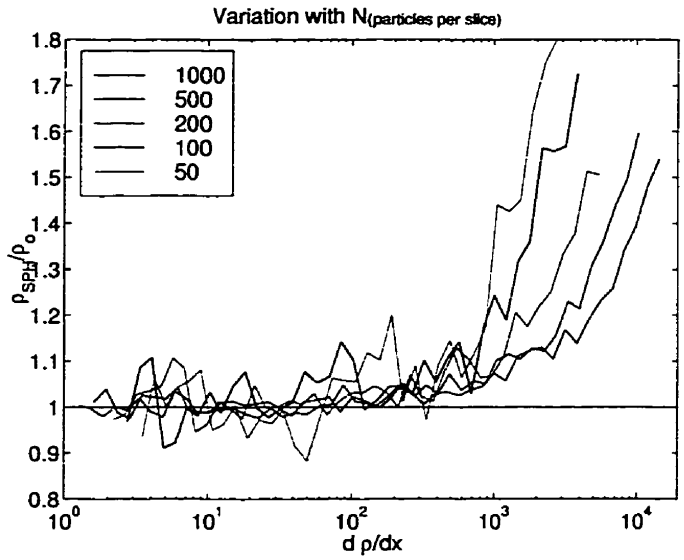


Figure 3.5: The error in the SPH density estimate as it varies with the number of particles per slice. Shown are the errors in the density estimate for sets of data as they vary with the gradient. The sets of data correspond to different numbers of particles per slice which are given in the legend.

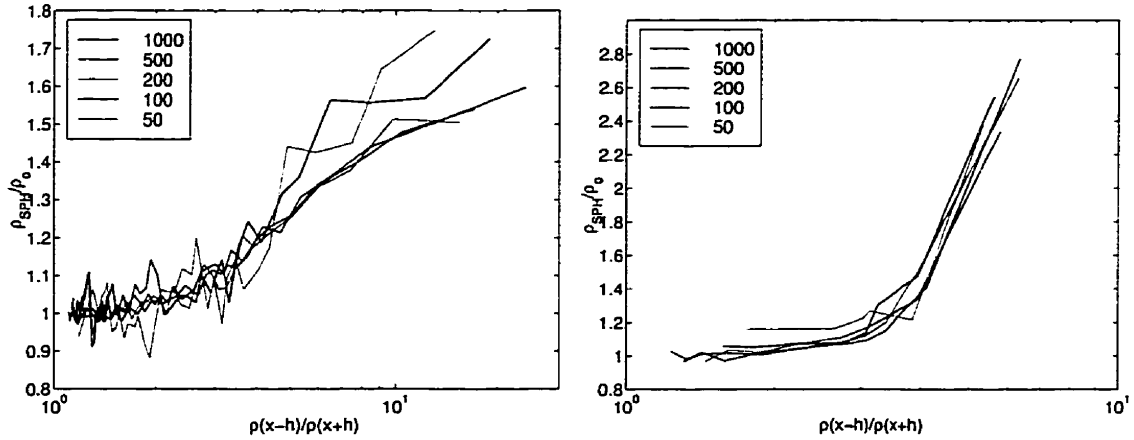


Figure 3.6: The error in the SPH density estimate *vs.* the density ratio over $2h$. Shown are the errors for sets of data with different slice density factors, F. On the left, $F = 1.1$ while $F = 1.5$ on the right. The abscissa gives the ratio of the actual gas density a distance h into the higher density regime to the density a distance h into the lower density regime. These distances, h, are not the same for the different curves but rather decrease with an increasing number of particles per slice (given in the legend).

F, can be defined as

$$F \equiv \frac{n(r-h)}{n(r+h)} \tag{3.16}$$

where n is the number density of particles, or ρ/m. The critical radius, r_c, below which the density estimates will be in error is related as

$$\frac{n(r_c - h(r_c))}{n(r_c + h(r_c))} = F_c. \tag{3.17}$$

The estimate $n(r+\delta r) \approx n(r) + \delta r \frac{dn}{dr}$ gives

$$\frac{n(r-h)}{n(r+h)} \simeq \frac{n(r) - h(r)\frac{dn}{dr}}{n(r) + h(r)\frac{dn}{dr}}. \tag{3.18}$$

If the density profile is assumed to take the form, $n(r) = n_o r^{-\alpha}$ then $dn/dr = -\alpha n_o r^{-\alpha-1}$ leading to

$$\frac{n(r-h)}{n(r+h)} \simeq \frac{n_o r^{-\alpha} + \alpha h(r) n_o r^{-\alpha-1}}{n_o r^{-\alpha} - \alpha h(r) n_o r^{-\alpha-1}} = \frac{1 + \alpha h(r) r^{-1}}{1 - \alpha h(r) r^{-1}} = \frac{r + \alpha h(r)}{r - \alpha h(r)}. \tag{3.19}$$

Using the relation between h, N_{SPH}, and n,

$$h(r) = \frac{1}{2}\left(\frac{3}{4}\frac{N_{SPH}}{\pi n(r)}\right)^{\frac{1}{3}}, \tag{3.20}$$

and Eq. 3.17, the value for r_c can be derived,

$$r_c^{1-\frac{\alpha}{3}} = \frac{\alpha}{2}\left(\frac{3}{4}\frac{N_{SPH}}{\pi n_o}\right)^{\frac{1}{3}} \frac{F_c + 1}{F_c - 1}. \tag{3.21}$$

In these studies, $N_{SPH} = 32$ and, as seen, $F_c \approx 3$, giving the approximate equation,

$$r_c^{1-\frac{\alpha}{3}} \simeq \frac{2\alpha}{n_o^{\frac{1}{3}}}. \tag{3.22}$$

This allows the derivation of an approximate minimum radius for a given cluster with a compression factor (*c.f.* overdensity) of δ_c at the critical radius, r_c. The compression

factor is defined as the density at the critical radius compared with the mean density in the volume. That is,

$$\delta_c \equiv \frac{\rho_c}{\langle \rho \rangle_V} = \frac{n_c}{\langle n \rangle_V}, \tag{3.23}$$

where the notation $\langle A \rangle_V$ refers to the volume-weighted mean of the parameter A. Since $\langle n \rangle_V = Res^3$, where Res is the number of nodes per side in the initial density distribution (*i.e.*, $N_{gas} = Res^3$),

$$n_c = \delta_c Res^3. \tag{3.24}$$

Recall that $n(r) = n_o r^{-\alpha}$, implying $n_o = \delta_c Res^3 r_c^{\alpha}$ giving,

$$r_c \simeq \frac{2\alpha}{\delta_c^{1/3} Res}. \tag{3.25}$$

For the maximum overdensity observed, $\delta_c = 10^6$, $r_c = 40\,\mathrm{kpc}$ for simulations with 64^3 particles. For $\delta_c = 10^3$, this increases to $r_c = 400\,\mathrm{kpc}$. Clearly, r_c is enclosing a constant number of particles. This minimum number of particle, $N_{min} = \frac{4}{3}\pi r_c^3 \delta_c Res^3$, when r_c from Eq. 3.25 is substituted, gives,

$$N_{min} \simeq 4(2\alpha)^3. \tag{3.26}$$

This number is the minimum number of gas particles, within r_c, required to properly resolve the density gradient given by α. For $\alpha = 2$, the required number is 250 but this number is sensitive to α and drops to 100 for $\alpha = 1.5$

These numbers, r_c and N_{min}, are useful for determining minimum smoothing lengths (via the minimum softening length in HYDRA) during the simulation and the minimum cluster size to be used in the analysis.

3.2 Cooling Near Steep Density Gradients

Large density gradients occur in the gas in cosmological simulations as a result of radiative cooling which leads to cold dense knots of gas within hot haloes. Investigations performed with the current HYDRA code have illuminated a disturbing 'supercooling' phenomenon associated with these knots of gas that produces an unphysical, dense, gaseous object.

Essentially, this phenomenon is a manifestation of SPH's inability to model arbitrarily steep density gradients. The density gradient between a cold clump of gas and the hot halo should, physically, be sufficient to keep the media from interacting with each other. The hot gas should cool according to the temperature and density of the hot gas. Only once it is cooled by this slower method should it interact with and accrete onto, the cold clump. However, as seen in Sec. 3.1, the density of the hot, low density gas is over-estimated in the presence of a dense clump of cold gas. This leads to a reduced cooling time ($t_{cool} \propto T^{1/2}/\rho$) for the hot gas which in turn leads to an accelerated accretion rate onto the cold clump and a steeper density gradient. The dense cold clump is prevented from further collapse by random motions within the resolution-determining gravitational softening length.

This effect is termed *overcooling*. It should not be confused with the overcooling problem (or 'cooling catastrophe') in simulations of galaxy formation.

In this section, toy models will be used to examine the cooling rates of the hot gas in a hot-halo-cold-clump system evolved under the variety of implementations of SPH described earlier. First, the halo-clump model system will be described. Then the overcooling phenomenon will be described in more detail with the use of variants of this toy model. An estimate of the variation in cooling rates due simply to initial particle positions is made. Finally, the results of the different implementations will be presented.

Cluster	$5 \times 10^{14}\,M_\odot$	$5 \times 10^{13}\,M_\odot$
R_{clump} (kpc)	20	9
m_{gas} ($10^{10}\,M_\odot$)	1	0.1
m_{dark} ($10^{10}\,M_\odot$)	9	0.9
ϵ (kpc)	20	10

Table 3.2: Cluster parameters. R_{clump} is the radius of the cold clump, ϵ is the gravitational softening length and m_{gas} and m_{dark} are the mass of a gas and dark-matter particle respectively.

	Cold clump	Halo gas	Halo dark-matter
N	500	4737	4994
T (K)	10^4	$2 \times 10^5 - 10^8$	N/A
ρ/ρ_c	2×10^6	$10^2 - 10^5$	$10^3 - 10^6$

Table 3.3: Cluster parameters in common. N is the number particles, T the temperature range and ρ/ρ_c the ratio of the density to the critical density.

3.2.1 Description of the halo-clump systems

To examine this phenomenon, core-halo systems were created, each consisting of a cold clump of gas surrounded by a hot-gas halo both embedded in a dark-matter halo. The dark-matter and hot-gas system was extracted directly from a cosmological simulation. The cold clump was created by randomly placing particles inside a sphere of size equal to the gravitational softening length and allowing this system to evolve to a relaxed state. The cold clump was then placed in the hot-gas and dark-matter system. Two core-halo systems – designed to resemble galaxy clusters – were created to test the effect of mass and linear scale dependence, with total masses $5 \times 10^{14}\,M_\odot$ and $5 \times 10^{15}\,M_\odot$. The parameters of the systems are listed in Tables 3.2 and 3.3.

Because the time-step criterion used in all of these versions makes no reference to the rate of change of the temperature nor to density changes (Thomas and Couchman 1992), it is possible that hot halo particles may not cool correctly as they accrete on

to the cold clump. An in-falling particle approaching the central cold clump is heated by the conversion of potential energy into thermal energy. If it cools rapidly enough it can accrete onto the central clump, but if the time step is too long it may receive too large a temperature jump and become spuriously hot. Hence, the normalisation factor, κ, for the time step affects the accretion rate appreciably. This factor, κ, simply scales the 'suggested' time step to the 'adopted' time step via $dt = \kappa dt_C$ where dt_C is set by the Courant conditions. From a series of tests it was determined that, for $\kappa > 0.5$, the shorter the time step, the greater the accretion rate, and the greater the total amount of accretion. There is convergence for $\kappa \leq 0.5$ so we adopt a value of $\kappa = 0.5$ for these cooling tests.

3.2.2 Testing the overcooling phenomenon

The nature of this overcooling problem is revealed by a series of tests involving a dense gravitational source in the centre of a halo of hot gas. These clump-halo systems are described in more detail below. In the first case, the dense knot onto which the hot gas will cool is a clump of cold gas. In the second test, the cold clump is replaced with collisionless matter and the hot gas is allowed to cool to form its own cold and dense knot. For the final test, all cold matter is turned into collisionless matter after it cools below a certain threshold (2×10^4 K). These tests are denoted as 'standard', 'collisionless' and 'conversion', respectively. If the hot gas did not interact with cold gas then these tests would all give the same result. The number of particles cooled over time for these three tests for the 5×10^{14} $M_\odot$ cluster is shown in Fig. 3.7.

The behaviour at early times ($< 2 \times 10^9$ yrs) is dominated by a sudden rise in the number of cold particles. This is due to the system equilibrating as the hot gas falls toward the high central density, increasing its density and cooling rate. The standard test clearly shows the fastest increase in cooling, thus displaying the overcooling phenomenon. The collisionless test shows a decreased rate of cooling until a sufficient

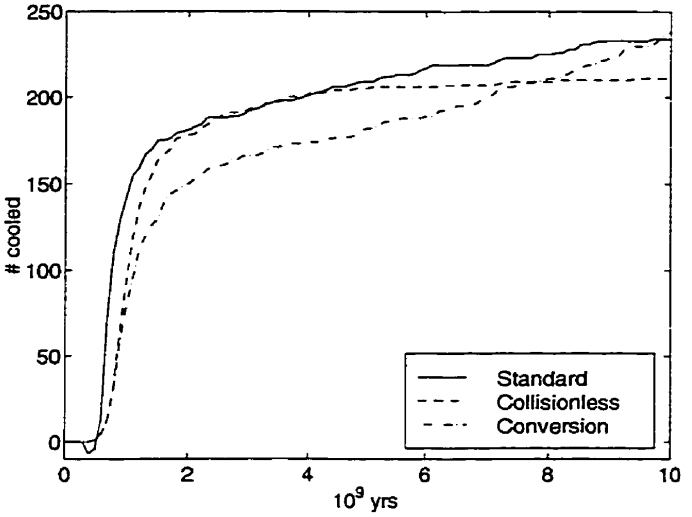


Figure 3.7: The number of cooled particles in the three experiments as a function of time. If the hot gas did not interact hydrodynamically with the cold dense clump in the simulations, these lines would overlap.

number of gas particles (approximately 30) have accreted in the central region to start the overcooling process. The cooling rate then 'catches up' to the standard case at which time both have depleted the local reserve of approximately 200 particles. For the conversion test, in which the cold, dense clump of gas required to initiate the overcooling phenomenon is never allowed to form, the cooling rate lags dramatically behind the other two tests.

At later times ($> 2 \times 10^9$ yrs), the results of the standard and collisionless tests diverge with the standard test displaying an excess amount of cooling due to its larger number of cold-clump particles (600 versus 200 for the collisionless test) and, consequently, its larger density gradient. As for the conversion test, the increasing central mass density of the halo, coupled with the absence of a central gas clump to provide pressure support, leads to a progressively increasing cooling rate which is not related to the overcooling phenomenon but rather the aphysical nature of the test.

The description of the overcooling phenomenon given here predicts a jump in the calculated SPH gas density for the hot particles at the point where the cold clump

falls within the limits of the SPH smoothing kernel. The behaviour of the gas density at the interface between the gas phases is illustrated in Fig. 3.8 and confirms this prediction. For the standard test, the smoothing process forces the density to rise very abruptly from the halo to the core, while for the conversion test the lack of a cold gas core removes this imperative. The slope of the log-log density profile makes a jump from about -2 exterior to the SPH kernel limit to about -4 interior to this limit. Only once the particles get within the smoothing length for the cold particles does the profile plateau. If the hot particles are not permitted to interact with the cold particles, as in the conversion test, then the density profile should flatten out within a distance equal to their kernel softening length, not steepen. Consequently in the standard test, particles outside the dense core, but within $2h_{hot}$, have a higher density and thus a higher cooling rate.

Due to the higher cooling rate for the hot gas in the standard test, once particles fall within $2h_{hot}$ an abrupt temperature decrease should result. This unphysical drop in temperature is clearly illustrated by comparing the temperature profile of the gas produced in the standard test with that produced by the conversion test (Figures 3.8 (a) and (b)). If the hot gas should not interact hydrodynamically with the cold, dense gas, then the test version should most accurately reproduce the temperature profile in the core region. For this test the gas is approximately isothermal except in the core, where the increased density has caused the gas to cool. This is in contrast to the abrupt drop in the temperature of the hot gas in the regular SPH implementation, coincident with the cold clump falling within the $2h$ distance from the hot gas particle. This distance marks the limit of the SPH smoothing kernel for these particles. Note that, in the conversion test, the central core is more extended than in the standard test, but remains within a sphere smaller than the hot gas smoothing radius.

Not surprisingly, the cooling time for the hot gas takes a sharp drop interior to

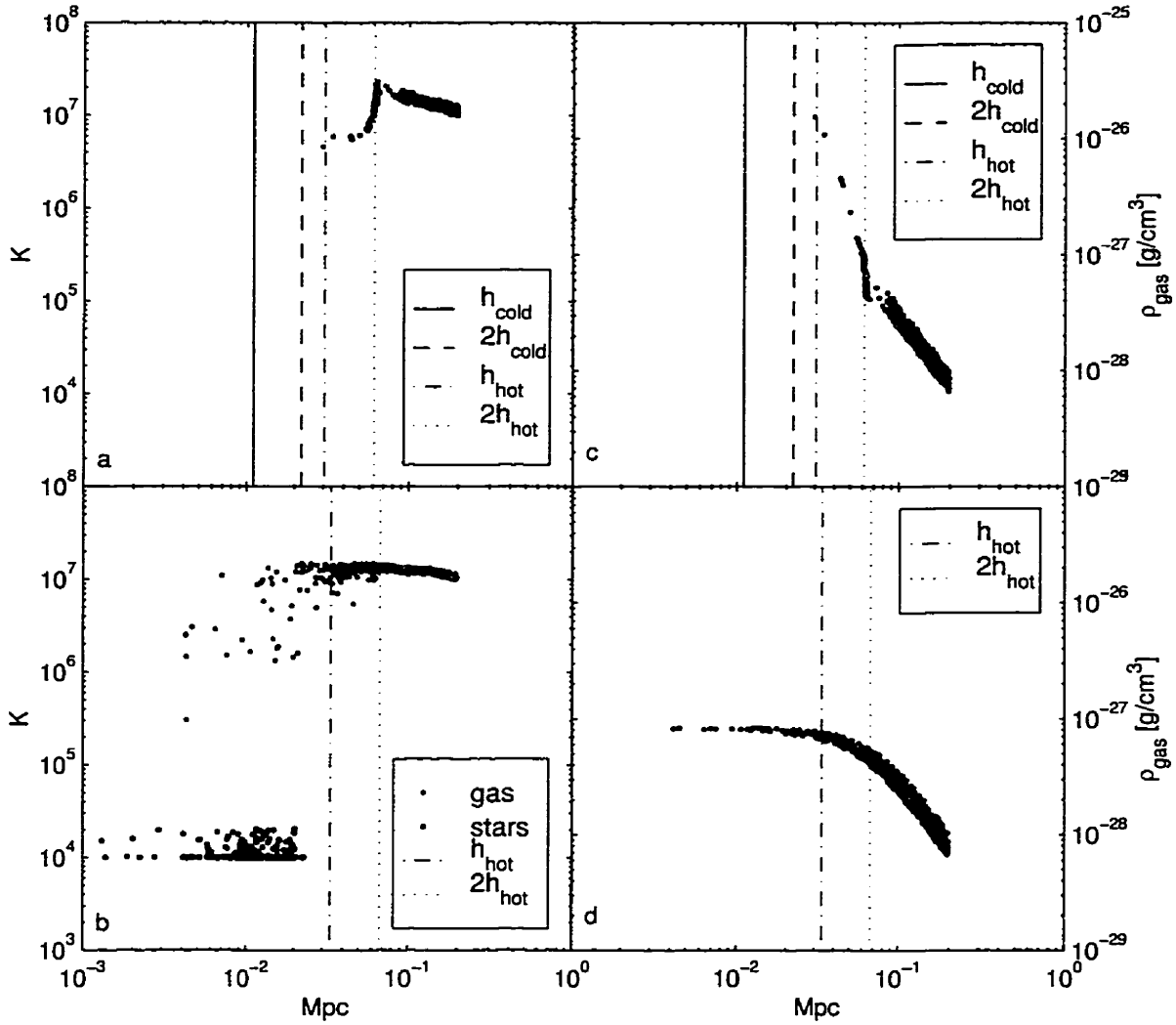


Figure 3.8: Temperature and density profiles for the standard test (a, c) and conversion test (b, d). The mean smoothing radii for both the hot and cold particles are represented by the vertical lines. The line for h shows the point internal to which the bulk of the kernel weighting is accumulated. The $2h$ line gives the outer limit of the smoothing. In panels a) and c), the inner pair of lines (left hand) are for the cold gas, while the outer pair are for the hot gas. In the lower panels, the cold particles are collisionless, and hence have no smoothing length. For these data, the initial conditions of the 5×10^{14} $M_\odot$ mass cluster were used. In the conversion test, the stars are plotted at the temperature to which they had cooled just before conversion to star particles.

the limit of the SPH softening kernel for this gas in the standard test (Fig. 3.9). In the conversion test, this is not observed. This is entirely consistent with the other evidence for the supercooling phenomenon seen previously.

The cooling rate of the gas is also affected by the virial temperature of the halo gas and, hence, the mass of the cluster. The overcooling for the two different mass clusters is compared in Fig. 3.10.

3.2.3 Variation due to initial particle positions

It is useful to know what is the typical variation in the cooling rate due simply to different particle distributions in halos with otherwise the same properties. This allows for a proper appreciation of what is a significant variation in the cooling rate. Fig. 3.11 indicates a variation of 40 in the number of particles that have cooled at any one time after 2×10^9 yrs. A similar test was performed with the collisionless model described in the previous section. The cooling rates produced by the various test versions of the code are essentially identical.

3.2.4 Results of the different SPH implementations

Experiments were run with all 12 different SPH implementations. The collisionless tests produced cooling rates which were essentially identical.

The variation in the overcooling rate among the implementations of the viscosity is given in Panel a) of Fig. 3.12. The cooling rates are not appreciably affected by the viscosity used. The exception is the one-sided implementation of the Monaghan viscosity (version 11) which produces 50% more cool particles over the time-span examined. This version uses an estimator for the density of the second particle based on the smoothing lengths and consequently has larger errors for the density. However, version 12 uses this viscosity with the kernel-averaging version of TC92 and it does not exhibit this behaviour. The inclusion of a shear-correction term in the artificial

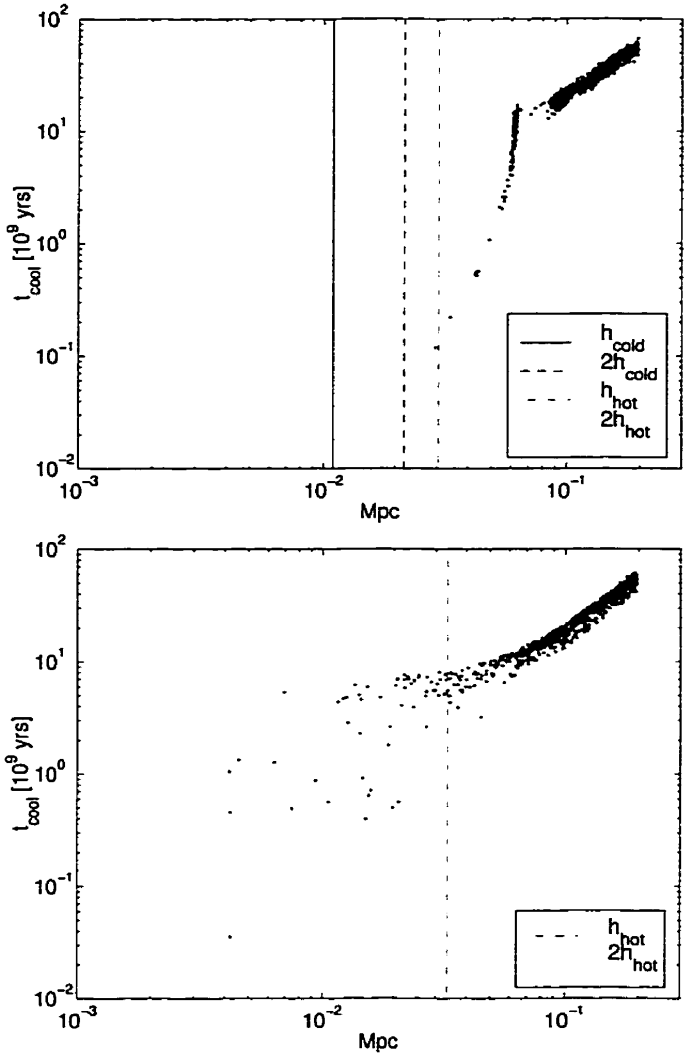


Figure 3.9: The cooling time for the gas particles around the cold clump as it varies with distance from the centre of the cold clump. The standard test produces the data for the upper plot, while the conversion test produces the data for the lower plot. See Figure 3.8 for an explanation of the vertical lines.

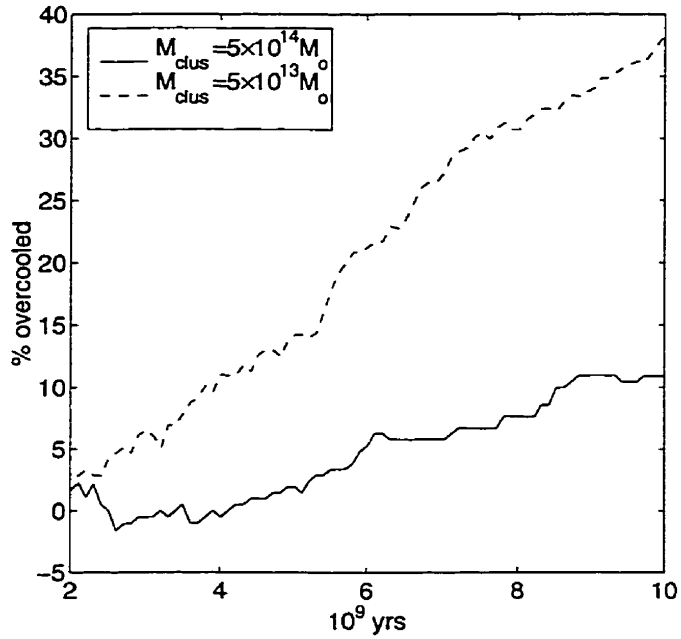


Figure 3.10: The amount of overcooled gas for two clusters of different mass. The ordinate is the percentage excess of cooled gas for a 'standard' run relative to the amount cooled in the corresponding 'collisionless' run.

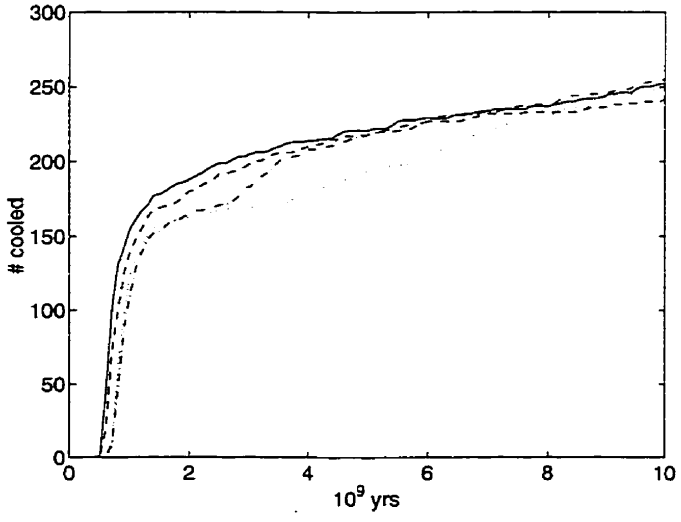


Figure 3.11: The number of cold particles ($T < 10^5$ K) in four different instantiations of the cold clump surrounded by the hot halo as they evolve in time. This reveals variation due to the finite number of particles imposed by the simulations.

viscosity (Fig. 3.12 b) has little effect on the cooling rate – as expected.

Since the overcooling effect is caused by the large difference in kernel sizes associated with the hot halo particles and the cold clump particles, it might be expected that the symmetrisation method has a role to play in determining the cooling rate. Consider the h-averaging schemes: the arithmetic mean is limited to having a minimum value of $h_{large}/2$, while the harmonic mean is zero if any particle interacts with another particle having $h = 0$. In practice, as is shown in Figures 3.12 c) and d), there is little difference amongst all symmetrisation schemes, with the exception of version 11. This version combines a pure gather kernel with the single-sided Monaghan artificial viscosity. This result is surprising in view of the comparatively 'normal' results for version 10, which differs in terms of the artificial viscosity, and version 12, which differs in terms of the symmetrisation.

3.2.5 Summary

All the versions of SPH we have tested exhibit overcooling and this effect should be seen as generic to the method itself. SPH will always experience difficulties modelling arbitrarily steep density gradients. The only implementation that stands out as performing poorly is version 11 which couples a one-sided implementation of Monaghan artificial viscosity with the TC92 symmetrisation procedure. When the TC92 symmetrisation is supplemented with kernel averaging, the performance is improved.

3.3 Drag

There is concern that the over-merging problem encountered in N-body simulations of clusters of galaxies is exacerbated in simulations which use SPH (see Frenk, Evrard, White, and Summers 1996). Excessive drag on small knots of gas within a hot halo will cause the knots to spiral inward into regions of stronger tidal forces where they may be disrupted (e.g., Moore et al. 1996).

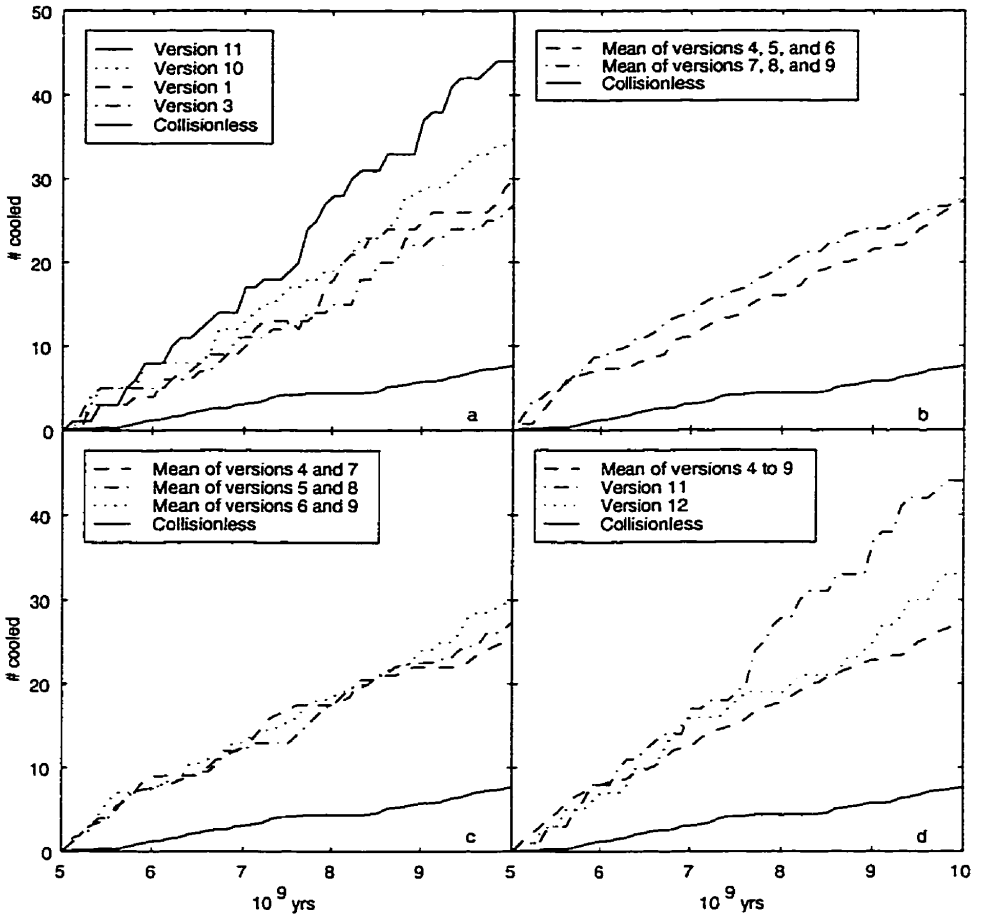


Figure 3.12: Variation with time in the number of cold particles ($T < 10^5$ K). Panel a) compares the different implementations of the artificial viscosity. The viscosities used are TC92 in original forms (dashed line) and in the more localised implementation (dot-dashed line), and Monaghan in its regular form (dotted line) and its one-sided version (upper solid line). Panel b) compares the cooling rate with the Balsara term (dot-dashed line) to that without (dashed line). Panel c) varies the symmetrisation scheme. The dashed line is the mean of those runs using arithmetically averaged values of the smoothing lengths. The same for those that use a harmonically averaged value of h are given by the dot-dashed line. If the kernels themselves are averaged, the dotted line is the result. Panel d) probes the symmetrisation of TC92 and its variant. The mean of the rates given in Panel c) (dashed line) representing the non-TC92 forms are compared with the results of the run with TC92 symmetrisation (dot-dashed line) and TC92 with kernel averaging (dotted line). These data are for the 5×10^{14} $M_\odot$ cluster. In all panels the amount of cooling produced in the run with an initially collisionless dense core is given by the lower solid line.

In this section, we will conduct a series of tests involving the various implementations of SPH on toy model systems involving a cold, dense clump passing through a hot medium. First, the toy model systems will be described including estimates of the deceleration rates produced independent of the SPH implementation. Then the results of the tests using the various implementations will be given.

3.3.1 Drag test model systems

To cover a variety of infall speeds, we examine the deceleration of a knot of cold gas in three velocity regimes: Mach 2, Mach 1, and Mach 1/3. The Mach 2 and Mach 1 tests differ in terms of the speed of the cold knot ('fast' versus 'slow') and not the temperature of the hot gas. The Mach 1/3 test uses the same clump velocity as the Mach 1 test, but is performed in hotter ('very hot') gas. Table 3.4 gives the details of the cold clump and hot gas phases. Clump characteristics are selected to loosely emulate a poorly resolved galaxy with no dark matter, while the hot gas media are typical of the intra-cluster medium.

The hot gas was prepared from an initially random placement of particles, and then allowed to relax to a stable state. The cold clump was created by randomly placing particles within a sphere of radius equal to the gravitational softening length. The cold clump was allowed to relax in the same manner as the hot gas, before combining the two systems. The cold clump is supported against further collapse by random motions of the gas particles within the gravitational softening length.

The Jeans length, R_J, for the hot gas phases is sufficiently large to ensure stability even in the presence of the perturbation from the cold clump. Consequently, dynamical friction should not be important. This conclusion was confirmed by passing a collisionless cold clump through the hot medium – it experienced negligible deceleration.

	Slow cold clump	Fast cold clump
ρ/ρ_c	1000	1000
T (K)	10^4	10^4
R (kpc)	50	50
N	100	100
m ($10^9\,M_\odot$)	1.7	1.7
V_o (km/s)	500	1000
	Hot gas	Very hot gas
ρ/ρ_c	10	10
T (K)	10^7	10^8
N	13000	13000
m ($10^9\,M_\odot$)	1.7	1.7
V_s (km/s)	500	1500
R_J (Mpc)	6	18

Table 3.4: The characteristics of the cold clumps and the hot media used in the drag tests. Given are the overdensity, ρ/ρ_c ($h_{100} = 1$), the temperature, T, the radius of the cold clump, R, the number of particles in the medium, N, the mass resolution of the medium, m, the initial velocity of the cold clump, V_o, the speed of sound in the hot medium, V_s, and the Jeans length for the hot medium, R_J. The simulation volume in all cases is $(5\,\mathrm{Mpc})^3$. The 'fast cold clump' was used in the Mach 2 runs in combination with the 'hot gas'. The Mach 1 runs used the 'slow cold clump' embedded in the 'hot gas'. The Mach 1/3 runs used the 'slow cold clump' in the 'very hot gas'.

The box length, 5 Mpc, was chosen so that the cold clump was well separated from its images (arising from the periodic boundary conditions employed) and would move across the box only once without encountering its own wake. As in Sec. 3.2.1, an appropriate value of the time-step normalisation parameter, κ, was found. For these tests, a value of $\kappa = 1.0$ is used.

3.3.2 Expected deceleration

An expected rate of deceleration may be approximated by considering a disc sweeping through a hot medium, collecting all matter it encounters. This would represent a maximum expected rate of deceleration if dynamical friction is ignored. The gas is sufficiently hot to prevent the wake collapse necessary for dynamical friction to work. The solution for the velocity, V, of such a system is given by $V(t) = l/(t - t_l)$, where l is a characteristic length given by $l = M/2\pi R^2 \rho_g$ and t_l is a characteristic time-scale given by $t_l = t_o - l/V_o$. Here, M is the mass of the disc at the start, R is the radius of the disc, and ρ_g is the density of the gas through which the disc is travelling. The clump starts with velocity V_o at time t_o. For the tests that use the slow clump, this estimate implies the final velocity should be 400 km/s. For the fast clump, the final clump velocity should be 670 km/s. These crude estimates indicate that hydrodynamical forces should indeed be important for the parameters being considered.

The importance of the artificial viscosity can be estimated by performing a simulation with an implementation of SPH for which the artificial viscosity has been disabled. Fig. 3.13 compares such a run with the viscosity-enabled implementation for the three scenarios: Mach 2, Mach 1, and Mach 1/3 cold clumps. The implementation of viscosity used here is the standard HYDRA viscosity (TC92). A striking result is apparent. In those cases for which shocks are important, namely the Mach 1 and 2 cases, the presence of the viscosity term actually decreases the drag. The

effect is greater for the higher velocity clump. At early times, the drag is greater with the viscosity term since an amount of gas must be accumulated ahead of the clump before the viscosity-less pressure term can cause an effect. After this early transitory phase, however, the lack of a viscosity term, which prevents interpenetration of particles through shocks, results in hot gas particles getting closer to the cold clump and consequently receiving a bigger momentum boost in the direction of travel of the clump.

Four separate realisations of the same initial conditions were evolved with the same version of the test code to look for variation due to randomness in the initial conditions. There is variation on the order of 10 km/s between the runs for the Mach 2 and Mach 1/3 scenarios, 4 km/s for the Mach 1 scenario.

3.3.3 Results of the SPH variants

The cold clump varies in final size among the different versions of the code. The cold gas in the versions with the artificial viscosity following the form of Monaghan collapses to a much denser knot than the other versions (Fig. 3.14 and Table 3.5). The mean particle separation for the clump in the runs using Monaghan viscosity is on the order of 4 kpc which is a third of the value for the runs using TC92 viscosity. Indeed, the localised version of TC92 produces a mean particle separation for the cold clump of 33 kpc. However, in all cases the knot size is still much less than the smoothing radius, so the total clump size appears to the other gas particles, when the gas forces are calculated, to be essentially the same (Table 3.6).

Compared with $\nabla \cdot \mathbf{v}$ viscosity, Monaghan viscosity in both the symmetric and single-sided forms leads to an increase in the damping of the velocity of the clump when used with the TC92 symmetrisation (Table 3.7 and Fig. 3.15). However, when Monaghan viscosity is used with TC92 symmetrisation, supplemented by kernel averaging, the deceleration becomes comparable to the $\nabla \cdot \mathbf{v}$ versions. The more localised

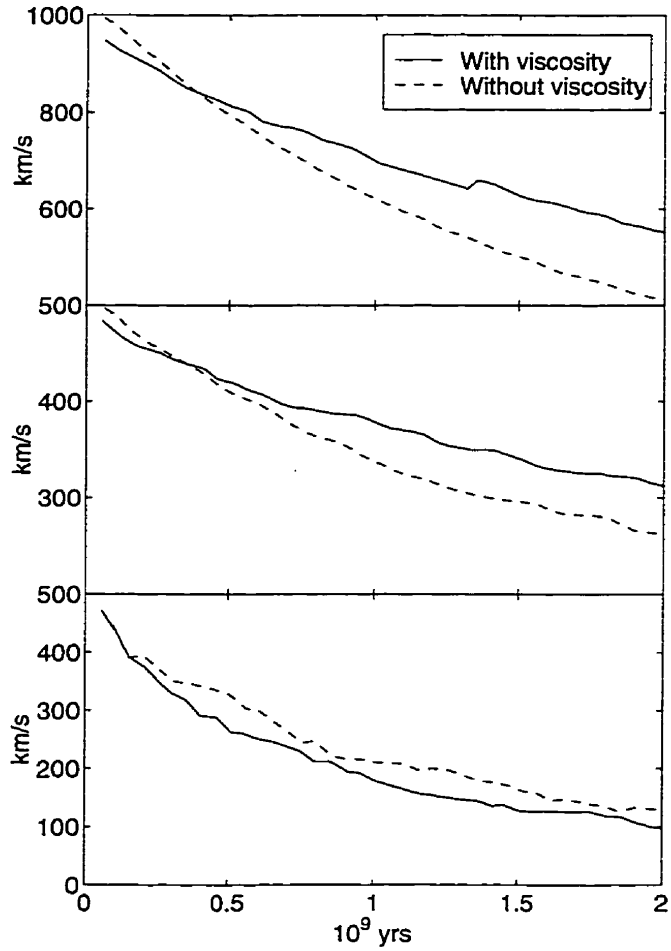


Figure 3.13: The significance of viscosity in the drag studies is demonstrated by a comparison of the velocity of the clump when it is evolved via an implementation with artificial viscosity enabled (solid line) and with it disabled (dashed line). The velocity of the cold clump as it changes with time is given for the Mach 2 runs (top), Mach 1 runs (middle), and the Mach 1/3 runs (bottom). The original version of the code was used for this test.

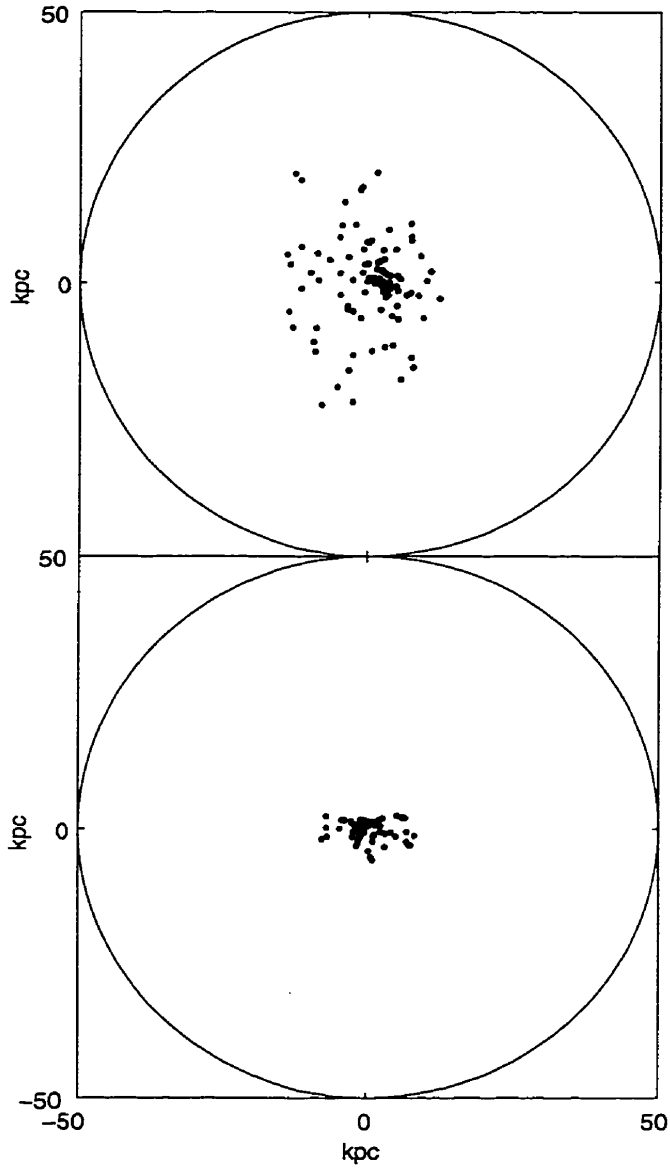


Figure 3.14: The particle distribution of the cold clump for versions 2 (top) and 5 (bottom). The circle denotes the smoothing radius, $h = 50\,\mathrm{kpc}$.

Version	$\langle r \rangle$(kpc)	$\langle h \rangle$(kpc)
1	12.76	51.65
2	12.94	51.65
3	35.66	52.07
4	3.93	51.51
5	3.89	51.51
6	2.94	51.62
7	4.55	51.56
8	4.98	51.57
9	3.72	51.62
10	3.18	51.66
11	3.27	51.66
12	3.32	51.61

Table 3.5: The sizes of the cold clumps compared with the smoothing lengths, h as they vary among versions of the code. The sizes of the clumps are estimated from the mean particle distance from the clump centre ($\langle r \rangle$). All measurements are for the final time step taken from the Mach 2 set of runs.

estimate of the $\nabla \cdot \mathbf{v}$ viscosity does little except in the Mach 2 set of runs, for which it increases the drag to match that of the Monaghan viscosity. The inclusion of a shear-correction term reduces the drag in the Mach 1/3 case as well as in the Mach 1 case when the clump velocity has dropped below Mach 0.8.

Use of either the arithmetic or harmonic average for h_{ij} produces less drag then any other symmetrisation method (Fig. 3.16) except the version with TC92 symmetrisation combined with kernel averaging. On their own, kernel averaging and the TC92 symmetrisation produce marginally higher deceleration, particularly at supersonic speeds.

3.3.4 Summary

All implementations produce more drag than should be expected. Among the symmetrisation methods, the tests favour (but cannot distinguish between) those that use the harmonic and arithmetic averages for calculation of h_{ij}. The artificial viscosity

	Cross section (kpc^2)	
Version	$\sigma > 1.5 \times 10^7\,M_\odot\,/\,kpc^2$	$\sigma > \max(\sigma)/2$
1	3982	4480
2	3988	4463
3	4162	4508
4	4215	4192
5	4214	4191
6	4209	4197
7	4202	4196
8	4200	4212
9	4210	4198
10	4210	4199
11	4210	4199
12	4210	4198

Table 3.6: The estimated cross-sections (σ) of the cold clumps produced by the different versions of the code. Projected column densities for the clumps were calculated in the direction of their motion. The second column is calculated using a column-density cutoff of $1.5 \times 10^7\,M_\odot\,/\,kpc^2$ which was found to approximate the position of the steep edge of the Gaussian-like distribution. The third column tabulates the full area with half the maximum. A condensed core is indicated if the cross-section from this column is much less than that of the column to the left. This tendency is consistent with Table 3.5. Note that a cross-section of $4200\,kpc^2$ corresponds to a disk radius of 36 kpc.

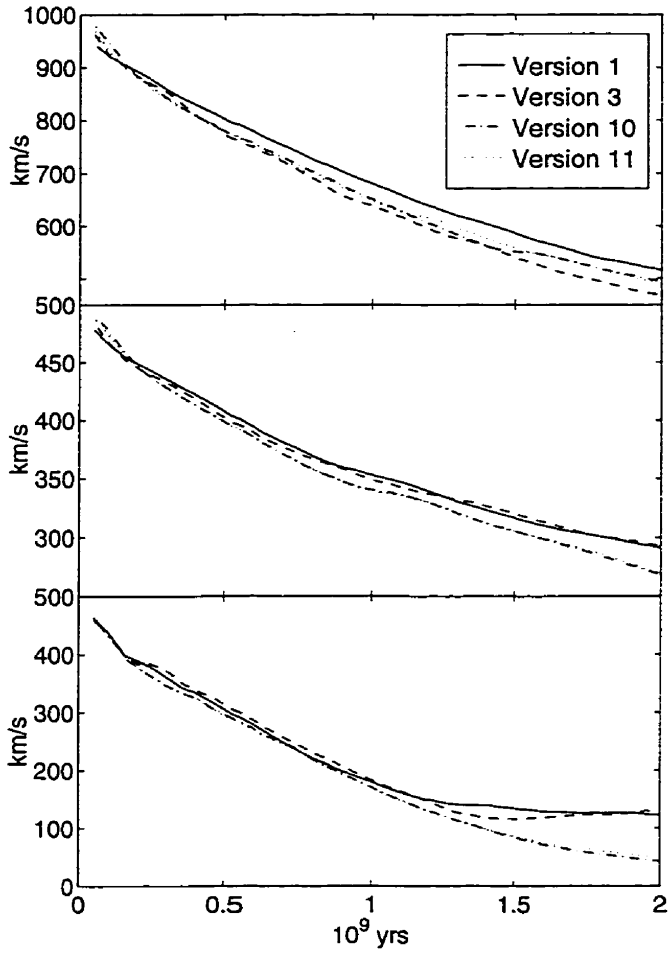


Figure 3.15: Variation of the cold-clump velocity with artificial viscosity type (no shear-correction term is included). In each panel the different lines distinguish different viscosity implementations: standard HYDRA viscosity (TC92), solid; localised TC92 viscosity, dashed; standard Monaghan viscosity, dash-dot; single-sided Monaghan viscosity, dotted. The panels are, from top to bottom; Mach 2, Mach 1 and Mach 1/3.

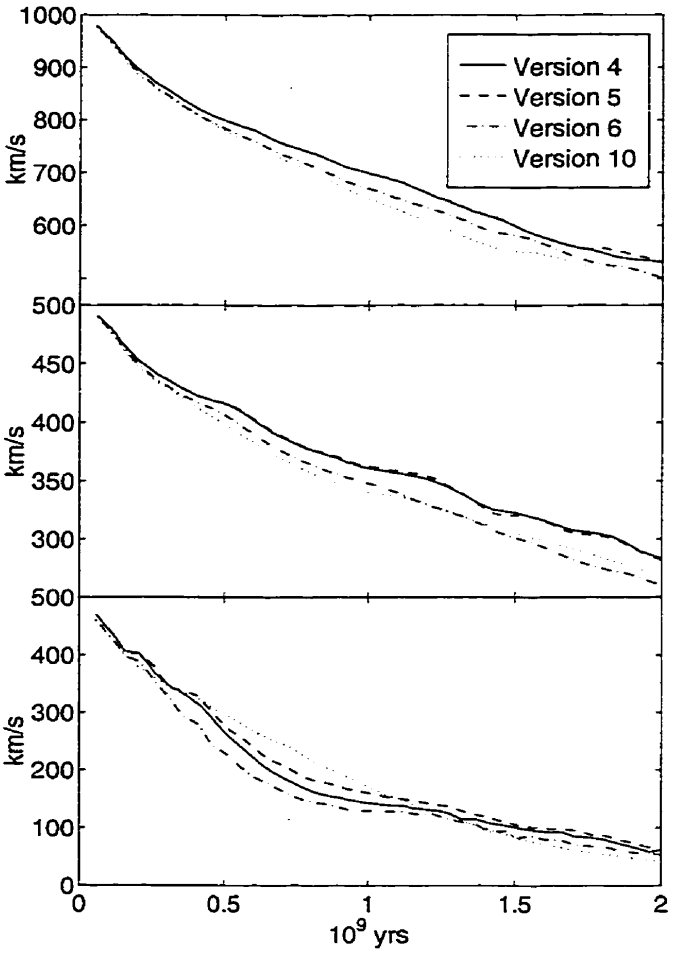


Figure 3.16: Variation of the cold-clump velocity with h-symmetrisation. The lines are: arithmetic average, solid; harmonic, dashed; kernel averaging, dot-dash; TC92 symmetrisation (version 10), dotted. The panels are, from top to bottom; Mach 2, Mach 1 and Mach 1/3.

Version	Mach 2	Mach 1	Mach 1/3
1	0.999 ± 0.002	1.004 ± 0.009	1.24 ± 0.10
2	0.991 ± 0.006	1.003 ± 0.012	1.29 ± 0.08
3	0.914 ± 0.006	1.009 ± 0.008	1.20 ± 0.16
4	1.018 ± 0.005	1.009 ± 0.008	0.78 ± 0.07
5	1.025 ± 0.009	1.004 ± 0.009	0.84 ± 0.07
6	0.984 ± 0.008	0.926 ± 0.013	0.67 ± 0.05
7	1.064 ± 0.007	1.060 ± 0.004	1.26 ± 0.02
8	1.045 ± 0.007	1.062 ± 0.008	1.34 ± 0.01
9	1.028 ± 0.006	0.979 ± 0.006	1.12 ± 0.05
10	0.957 ± 0.005	0.951 ± 0.007	0.58 ± 0.07
11	0.956 ± 0.004	0.955 ± 0.006	0.62 ± 0.06
12	1.018 ± 0.003	1.038 ± 0.009	1.06 ± 0.10

Table 3.7: The relative final velocities of the cold clumps. Given is the mean relative velocity of the cold clumps over the final 0.5×10^9 yrs normalised by the mean velocity of all the cold clumps in that velocity regime.

actually *decreases* the drag by reducing the interpenetration of hot gas particles into the halo of the cold clump. However, the tests are not sensitive to variations in the form of the artificial viscosity. The shear-correction term for the artificial viscosity lowers the drag at sub-sonic speeds but does little at sonic and super-sonic speeds. The Monaghan viscosity coupled with the TC92 symmetrisation performs poorly but it is not clear why this combination does so.

Chapter 4

SIMULATIONS

4.1 Numerics

The N-body $AP^3M - SPH$ code, HYDRA(Couchman, Thomas, and Pearce 1995) was used for all simulations. This is a multi-level Adaptive Particle-Particle Particle-Mesh (AP^3M) N-body code with gas dynamics simulated by the Lagrangian Smooth Particle Hydrodynamics (SPH) method (Sec. 3.0.1). The gravity is calculated using a particle-mesh scheme for the large scale gravitational fields. Short-range forces are calculated by summing particle-particle forces. Isolated regions of high number density have sub-grids adaptively placed around them, allowing the efficient particle-mesh method to be used locally. As such, it is well suited for the large dynamic range of scales involved in cluster studies. A comparison of the leading hydrodynamic codes, including HYDRA, designed for studying cosmological scenarios finds general consistency among the codes (Frenk and *et al* 1998).

4.2 Cosmology

All simulations assumed the flat cosmology given in Table 4.1. The matter was evolved in a box with $40h^{-1}$ Mpc sides in co-moving coordinates, with the Hubble expansion constant, $H_o = h100\,\mathrm{km\,/\,s}$. This permits the formation of a sufficient number of

H_o	$65\,\mathrm{km\,s^{-1}\,Mpc^{-1}}$
$\Omega_{DarkMatter}$	0.9
Ω_{Gas}	0.1
Λ	0
Power law index	-1
σ_8	.935

Table 4.1: Properties of the assumed cosmology.

clusters for statistical purposes. A box of larger dimensions would impose too great a penalty on the resolution of the simulation. The simulation had periodic boundary conditions.

Cooling is neglected. The cooling time for the bulk of the cluster gas is estimated to be well over the age of the universe. The peak luminosity is on the order of $10^{44}\,\mathrm{erg\,s^{-1}}$. The thermal energy found in the gas of a system containing $10^{14}\,\mathrm{M_\odot}$ of gas at 5 keV is on the order of $10^{63}\,\mathrm{erg}$ giving a cooling time, $t_{cool} = 10^{19}\,\mathrm{s} \approx 3 \times 10^{11}\,\mathrm{yrs}$. There is insufficient resolution due to the limit set by the gravitational softening parameter to properly model the cooling flows which are inferred in the inner 200 kpc of clusters (Allen and Fabian 1997). It would require a great deal more resolution in order to model these flows correctly owing to the overcooling induced by the inability of SPH to properly calculate the gas densities in the cores of clusters (see Sec. 3.2). As well, when modelled with sufficient resolution, the cooling flows simulated are much greater than those observed, likely due to a lack of feedback mechanisms such as supernova energy input which reheats the gas (Suginohara and Ostriker 1998). In any case, the amount of gas inferred to have cooled in these flows is on the order of $10^{11}\,\mathrm{M_\odot}$, which is not significant to the larger scale distribution of gas in the cluster.

4.3 Initial Conditions

The initial density perturbations were established by displacing the particle positions from a uniform cubic grid using, in the standard way, the Zel'dovich approximation for growth of density perturbations in the linear regime (Zel'dovich 1970). This method first creates a representation of the density perturbations in Fourier k-space using a supplied power spectrum form supplemented with Gaussian random fluctuations. This is then transformed into an initial density field. The gravitational forces felt by the set of particles, distributed at the nodes of the uniform mesh, by this density field are calculated. The particles are then displaced in the direction of their respective forces an amount proportional to the force. This produces a distribution of particles with a density field following the previously calculated density field.

The initial redshift for the simulations is $z_{initial} = 75$. This was chosen to keep the maximum displacement incurred during the establishment of the density field to less than 1/2 the initial grid spacing. This keeps negligible the errors incurred in using the linear Zel'dovich approximation to a non-linear system. The initial power spectrum of the density fluctuations follows a power-law of $n = -1$, $P(k) \propto k^{-1}$.

In order to ascertain the effects hierarchical clustering has on the matter, runs with initial high-spatial-frequency density perturbations were compared to runs in which these density perturbations were suppressed. The perturbation suppression was accomplished via two methods. In one case, the initial density field was convolved with a tophat. This modification to the power spectrum takes the form $P'(k) = P(k)W(kr_{smooth})$ where $W(x)$ is the tophat window function given by,

$$W(x) = \frac{3(\sin x - x\cos x)}{x^3}. \tag{4.1}$$

A choice of $r_{smooth} = 7h^{-1}$ Mpc for the smoothing length was selected to suppress those perturbations of spatial size less than the size of a cluster formation region. Conse-

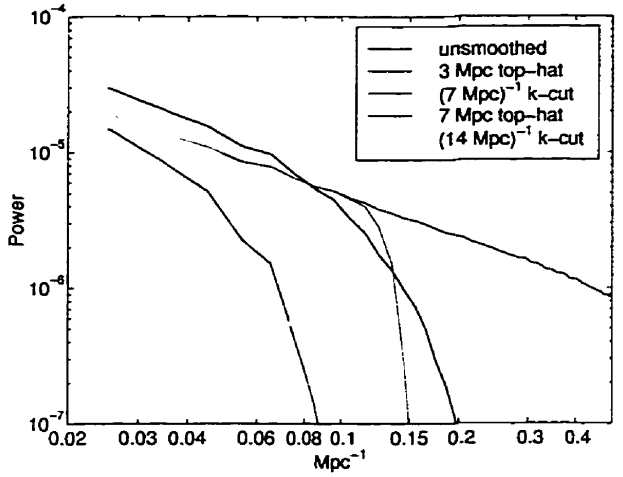


Figure 4.1: The initial power spectra of the simulations.

quently, the clusters in this scenario form from the uniform collapse of structures of a size on the scale of or larger than those that formed the clusters seen at the present. To span the regimes, a set of initial conditions smoothed with $r_{smooth} = 3h^{-1}\,\mathrm{Mpc}$ was also evolved. The second method for smoothing the initial power spectrum is a low-bandpass filter applied to the power spectrum of the density-perturbations before the particles are displaced. This filter is of the form

$$P'(k) = P(k)e^{-\left(\frac{kr_{smooth}}{2\pi}\right)^n} \tag{4.2}$$

where n is some positive integer which controls the steepness of the cut-off. A large value arbitrarily set to $n = 16$ was used here. Two sets of initial conditions for which $r_{smooth} = 7h^{-1}\,\mathrm{Mpc}$ and $14h^{-1}\,\mathrm{Mpc}$ were created using this method. The power spectra for the initial conditions created with the top-hat smoothing as well as the frequency cut-off are illustrated in Fig. 4.1.

The first minimum in $W(x)$ is at $k \approx \frac{5}{7}\frac{2\pi}{r_{smooth}}$. Hence, tophat smoothing over a radius of $7h^{-1}\,\mathrm{Mpc}$ removes more power at frequencies $k < 2\pi/r_{smooth}$ than the frequency-cutoff filter. The significance of this is made apparent by noting that the frequency-cutoff filter is essentially a convolution in real space of the density

distribution with the window function, $W(x)$. The $W(x)$ function can crudely be thought of as a top-hat with a radius given by the position of the first minimum. This implies a frequency cut-off of $\frac{2\pi}{7h^{-1}\,\mathrm{Mpc}}$ is roughly comparable to smoothing with a top-hat of radius $5h^{-1}\,\mathrm{Mpc}$.

Since the top-hat smoothing removes power at all frequencies except 0, the power on the scale of $8h^{-1}$ Mpc was increased to maintain the same RMS fluctuations (σ_8) for all sets of initial conditions. This was not done for the $(14h^{-1}\,\mathrm{Mpc})^{-1}$ smoothing, as this smooths over a larger radius than $8h^{-1}$ Mpc nor was it done for the $7h^{-1}$ Mpc top-hat filter for which first minimum in k-space is too close to the scale of $(8h^{-1}\,\mathrm{Mpc})^{-1}$.

The effect of two-body interactions is minimised by the gravitational softening parameter, ϵ. These interactions heat the gas via an exchange of energy between the dark matter and gas phases during short-range encounters of pairs of these particles (Steinmetz and White 1997). If the mean interparticle spacing is on the order of or greater than ϵ, then the time, t_{2-body}, required to heat the gas to its temperature, T, via this process is given by,

$$\left(\frac{t_{2-body}}{10^{10}\,\mathrm{yrs}}\right) = 160\frac{\left(\frac{T}{10^6\,\mathrm{K}}\right)^{\frac{3}{2}}}{\left(\frac{\ln\Lambda}{5}\right)\left(\frac{m_{DM}}{10^{10}\,\mathrm{M}_\odot}\right)\left(\frac{m_{DM}}{m_{gas}}\right)^{\frac{1}{3}}\left(\frac{\rho_{DM}}{\rho_c}\right)h^2} \tag{4.3}$$

where $\ln\Lambda$ is the Coulomb logarithm (3-7 for most simulations), m_{DM} and m_{gas} are the respective masses per particle for the dark matter and gas, ρ_{DM} is the dark matter mass density, and ρ_c is critical density. Without a softening term in the gravitational forces, this process would heat the gas in the centre of simulated clusters to $10^7\,\mathrm{K}$ in 0.5×10^9 yrs. However, in the dense cores, where this process is most significant owing to the dependency, $t_{2-body} \propto \rho_{DM}^{-1}$, there are on the order of 60 particle within a softening length, ϵ, of any other particle.

To examine the effects of resolution, the runs were performed at two resolutions.

	# of particles		r_{smooth}	ϵ
	gas	dark	(h^{-1} Mpc)	(h^{-1} kpc)
unsmoothed	64^3	64^3	0	20
3 Mpc top-hat	64^3	64^3	3	20
$(7\,\mathrm{Mpc})^{-1}$ k-cutoff	64^3	64^3	7	20
7 Mpc top-hat	64^3	64^3	7	20
$(14\,\mathrm{Mpc})^{-1}$ k-cutoff	64^3	64^3	14	20
low-res. unsmoothed	32^3	32^3	0	40
low-res. 7 Mpc top-hat	32^3	32^3	7	40

Table 4.2: Properties of the simulations. Given are the number of each type of particle, the effective smoothing radius, r_{smooth}, and the gravitational softening length, ϵ. The length of a side of the simulation volume is $40h^{-1}$ Mpc for all simulations.

In one set, 64^3 particles of each of gas and a collisionless dark matter component were evolved. In the other set, 32^3 particles of each were used. The gravitational softening was $0.02h^{-1}$ Mpc and $0.04h^{-1}$ Mpc for the respective runs. The details of the simulations are given in Table 4.2. The difference in structure at $t = 1$ (*i.e.*, the present) is clearly illustrated in Fig. 4.2.

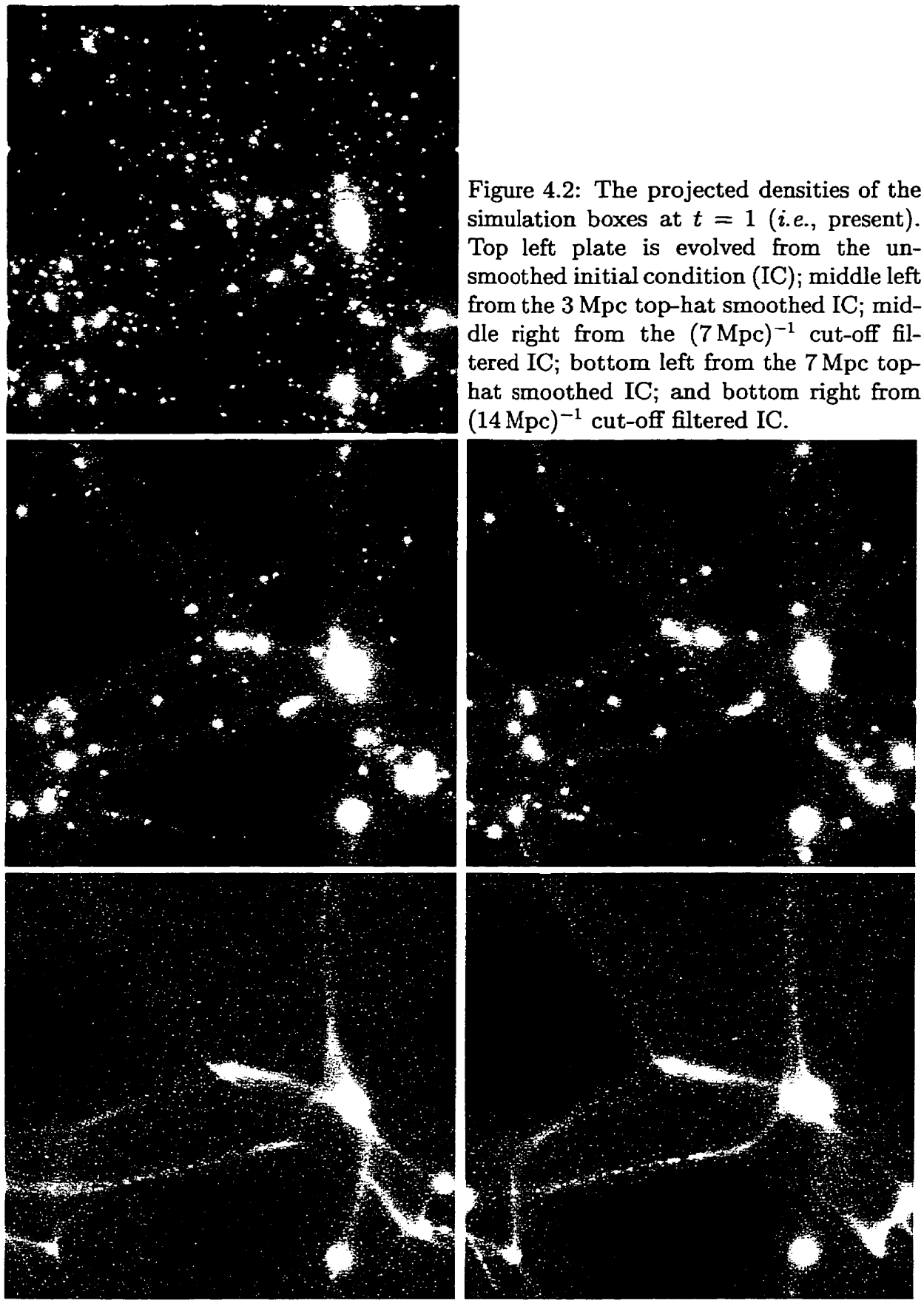


Figure 4.2: The projected densities of the simulation boxes at $t = 1$ (*i.e.*, present). Top left plate is evolved from the unsmoothed initial condition (IC); middle left from the 3 Mpc top-hat smoothed IC; middle right from the $(7\,\mathrm{Mpc})^{-1}$ cut-off filtered IC; bottom left from the 7 Mpc top-hat smoothed IC; and bottom right from $(14\,\mathrm{Mpc})^{-1}$ cut-off filtered IC.

Chapter 5

ANALYSIS

5.1 Cluster Selection

Clusters were selected by a tomographic deprojection method. The gas distribution, extended over its smoothing radius, was projected onto three planes: the x-y, the x-z, and the y-z. The resolution of these projections is $\approx 100\,\mathrm{kpc}$. Peaks were found in these three planes which were then tomographically deprojected to recover the (x,y,z) coordinates of the density maximums. The deprojection method involves the following steps:

1. Match the peak positions in the x domain (within some tolerance) for the x-y and x-z projections to find a list of possible y-z positions.

2. Match the y-z positions postulated from the previous step with the y-z positions found from the peaks in the y-z projection.

3. For each of the matches, use the x position from step 1) to get an x-y-z position.

This method is comparable to projecting the gas densities onto a 3-dimensional mesh of resolution $(boxsize)/L$, but requires $3 * L^2$ elements of information instead of L^3

which, beyond the memory requirement savings, speeds up the peak-finding algorithm.

Cluster searching using the friends-of-friends method (FOF V1.1) did not give satisfactory results. This method links together particles closer than a distance given by a linking length and then associates to a particle all its neighbour particles as well as their neighbours and so on. The association is then called a cluster. Close clusters are often connected by bridges allowing the association to 'percolate' through. As such, the grouping of cluster members was found to be too sensitive to the linking length parameter. For a choice of a linking length suitable for the largest clusters, the smallest clusters were not properly recovered. Another popular routine, SKID [1] (based on DENMAX), was tested. This routine 'freezes' the particles, then allows them to gravitate together to form tighter groups, essentially severing the bridges between clusters. A `fof` procedure is then allowed to create the list of asociations. Though the routine DENMAX performed better than FOF, particularly for the 2×32^3 simulation, the amount of CPU time required for the high-resolution simulation was unacceptable, taking days instead of hours.

The centres of the density peaks are further refined by cutting out a small sphere of particles, centred on the estimated density peak, and finding the mean position of the particles with the highest densities calculated previously by the SPH algorithm.

Overdensity radii, $R_{\bar{\delta}}$, were calculated for the clusters for overdensities of $\bar{\delta} \equiv \frac{\rho}{\rho_c} =$ 200 and 500. An overdensity radius is simply the radius from the centre of a cluster within which the ratio of the mean density to the critical density of the universe is equal to some value, referred to as the overdensity. Though spherical symmetry is not required, the derivation holds more relevance for spherically symmetric systems. The expected radius of virialization derived from analytic models involving simple

[1] see http://www-hpcc.astro.washington.edu/tools/SKID/

	$N_{clusters}$	$M_{min}[10^{15}\,M_\odot]$	$M_{max}[10^{15}\,M_\odot]$
unsmoothed	100	0.05	1.52
3 Mpc top-hat	48	0.05	2.85
$(7\,\mathrm{Mpc})^{-1}$ k-cutoff	48	0.05	1.75
7 Mpc top-hat	8	0.03	1.95
$(14\,\mathrm{Mpc})^{-1}$ k-cutoff	6	0.15	2.27
low-res. unsmoothed	3	0.54	0.70
low-res. 7 Mpc top-hat	5	0.40	1.88

Table 5.1: Results of the cluster search. Given for each run is the number of clusters found, $N_{clusters}$, and the range of masses of the clusters, M_{min} and M_{max}.

spherically symmetric collapse corresponds to that of an overdensity of 178 (*i.e.*, R_{178}). However, the overdensity radius, $R_{\bar{\delta}}$, is not particularly sensitive to $\bar{\delta}$. The overdensity within the calculated overdensity radius is accurate to a factor of N^{-1} where N is the number of particles. With the overdensity radii, there is a corresponding overdensity mass, $M_{\bar{\delta}}$, which is the mass contained within $R_{\bar{\delta}}$.

A lower limit to the size of the clusters was set by the requirement that each cluster, within the overdensity radius of R_{500}, have at least 300 gas particles and 300 dark matter particles. This ensures the densities are calculated correctly (see Sec. 3.1.2).

The details of the sets of clusters found using this method are given in Table 5.1.

A similar procedure was done using the dark matter mass distribution. The tomographic deprojection step for finding the clusters first required SPH-like density estimates be made for the dark matter and appropriate smoothing lengths be calculated in order to project the masses onto the x-y, x-z, and y-z planes. Little difference was found in the positions or number of the final clusters selected.

5.2 Cluster Profiles

Profiles of various parameters were calculated for the clusters by summing the con-

tributions of the particles falling in radial bins centred on the clusters. The bins were separated exponentially. Found for each of the dark matter and gas components were the number of particles, the volume-weighted mean density, and the mass-weighted mean thermal energy.

The thermal noise of the SPH particles provides a negligible contribution (1%) towards the thermal energy of gas particles in the halos. This contribution is ignored in all cases.

Mean profiles were calculated using the radially binned profiles described previously. The individual cluster profiles were scaled radially by the overdensity radius, R_{200}, as well as weighted by their respective cluster mass. For the density profiles, this weighting was found to have only a small effect, however, validating the use of mean profiles.

Density and temperature profile forms were fit to either the mean profiles or to the particle distributions of density or temperature *vs.* radius. Details of the fits are given in Sections 7.2, 8.3 and 8.4. In all cases, the fits were found by minimizing the χ^2 with the variance derived from the distribution of values in radial bins. Again, the distances were scaled by the overdensity radius, R_{200}. A profile fit was made to all the points in a given simulation.

It is the SPH-calculated density that is used when the particle distribution of density *vs.* radius is fit. Both the dark matter and gas particles had their densities calculated more precisely by performing an SPH summation over a value for the smoothing parameter, h, such that $2h$ encompasses exactly $N_{SPH} = 32$ particles. For the gas, this primarily affects the low and high density regimes. The densities were calculated without the self-density term (see Sec. 3.1.1) to provide a volume-weighted average local density at each of the particle positions. A comparison of

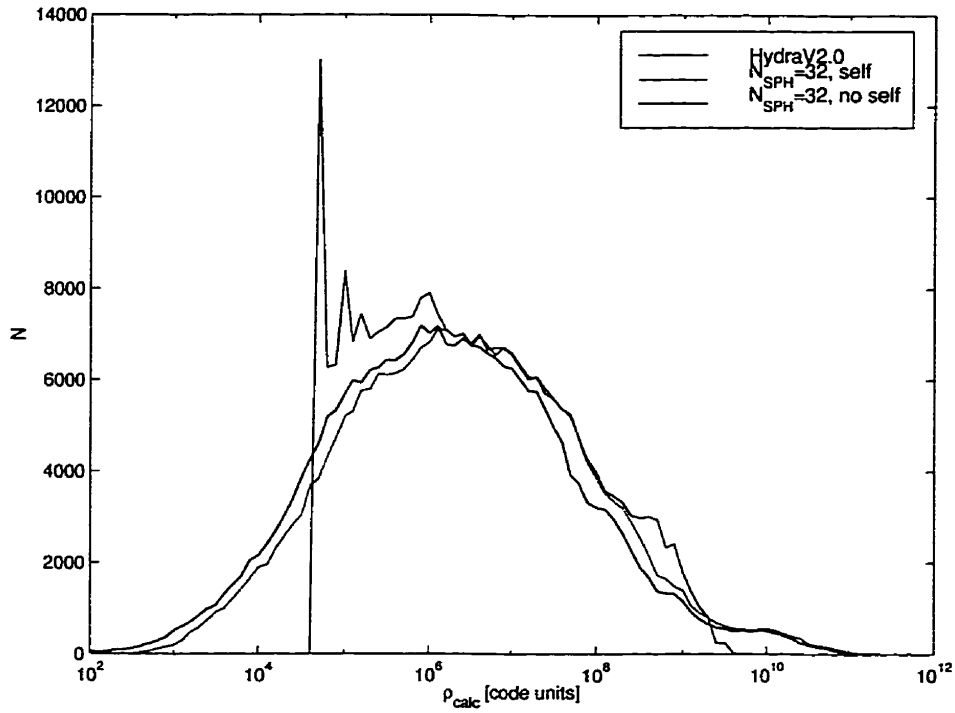


Figure 5.1: The distribution for the gas density calculated explicitly *vs.* that calculated by HYDRA (black line). The explicit calculations used 32 neighbours and were done using the self-density term (red) and without the self-density term (blue).

the densities calculated in this fashion with those that are found by HYDRA is given in Fig. 5.1. With the self-density term, the explicit method agrees with HYDRA for over two and a half orders of magnitude. HYDRA has limits on the size of h which leads to the discrepancies at low and high densities. The lower limit is set to improve computational efficiency; large values of h require neighbour searches over large volumes of the box even though the low-density gas with which they are associated is not relevant to most studies. The discrepancy at low densities is large, but the gas affected is found primarily in the voids away from the regions of interest. The upper limit is set to prevent over-collapse which is unwanted since both the physics at high densities is not complete and, as seen in Sec. 3.1.2, SPH cannot calculate densities correctly in large density gradients to arbitrarily small radii.

Chapter 6

THE HYDROSTATIC STATE OF THE CLUSTERS

The scaling law analysis in Chapter 8 assumes that the baryonic matter in the clusters is in hydrostatic equilibrium. To answer whether or not this is justified, a comparison of the actual pressure profiles for the clusters will be compared to the calculated pressure profile expected for a hydrostatic gas in the cluster's gravitational well.

6.1 A Problem with Binned Data

Consider the equation of state for a perfect gas, which relates the pressure P, with the gas density, ρ_g, and gas temperature, T:

$$P = \frac{\mathrm{k}}{\mu m_u}\rho_g T \tag{6.1}$$

with k, μ, and m_u being Boltzmann's constant, the mean-molecular weight, and the atomic mass number, respectively. We will use the notation $\langle A \rangle_V$ to represent the volume-weighted mean of the quantity A, and $\langle A \rangle_M$ to represent the mass-weighted

mean. In the case presented here, these will always be in a shell centred about the cluster, so $\langle A \rangle_\alpha$ (where $\alpha = V$ or M) can be taken to represent $\langle A \rangle_\alpha(r)$. Then the mean pressure in a shell is given by

$$\langle P \rangle_\alpha = \frac{\mathrm{k}}{\mu m_u} \langle \rho_g T \rangle_\alpha. \tag{6.2}$$

It is instructive to see how $\langle \rho_g T \rangle_\alpha$, $\langle \rho_g \rangle_\alpha$, and $\langle T \rangle_\alpha$ are related. Take for the i-th of n particles in the shell, $\rho_{gi} = \langle \rho_g \rangle_\alpha + \delta\rho_{gi}$ and $T_i = \langle T \rangle_\alpha + \delta T_i$, then

$$\begin{aligned}
\langle \rho_g T \rangle_\alpha &= \langle (\langle \rho_g \rangle_\alpha + \delta\rho_{gi})(\langle T \rangle_\alpha + \delta T_i) \rangle_\alpha \\
&= \langle \langle \rho_g \rangle_\alpha \langle T \rangle_\alpha + \langle \rho_g \rangle_\alpha \delta T_i + \langle T \rangle_\alpha \delta\rho_{gi} + \delta\rho_{gi}\delta T_i \rangle_\alpha \\
&= \langle \rho_g \rangle_\alpha \langle T \rangle_\alpha + \langle \rho_g \rangle_\alpha \langle \delta T_i \rangle_\alpha + \langle T \rangle_\alpha \langle \delta\rho_{gi} \rangle_\alpha + \langle \delta\rho_{gi}\delta T_i \rangle_\alpha \\
&= \langle \rho_g \rangle_\alpha \langle T \rangle_\alpha + \langle \delta\rho_{gi}\delta T_i \rangle_\alpha
\end{aligned} \tag{6.3}$$

For $n \to \infty$, $\langle \delta\rho_{gi}\delta T_i \rangle_\alpha = 0$ only if ρ_{gi} and δT_i are uncorrelated. This is unlikely for any gas, let alone gas in a cluster. If the shells themselves are in pressure equilibrium, *i.e.* $P_i = \langle P \rangle_\alpha$, then $\delta\rho_{gi}$ and δT_i will be anti-correlated, and $\langle \rho_g T \rangle_\alpha < \langle \rho_g \rangle_\alpha \langle T \rangle_\alpha$. If the matter in the shells has reached its state adiabatically, then $\delta\rho_{gi}$ and δT_i will be positive-correlated, and $\langle \rho_g T \rangle_\alpha > \langle \rho_g \rangle_\alpha \langle T \rangle_\alpha$. In the case of spherical symmetry, clearly $\delta\rho_{gi} = \delta T_i = 0$, and $\langle \rho_g T \rangle_\alpha = \langle \rho_g \rangle_\alpha \langle T \rangle_\alpha$.

If there is not spherical symmetry, as in the case of ellipsoidal isobars which would be found in an ellipsoidal gravitational potential, then $\delta\rho_{gi} = f_\rho(\theta, \phi)$ and $\delta T_i = f_T(\theta, \phi)$. If the gas is isothermal, then $\delta T_i = 0$ which gives $\langle \rho_g T \rangle_\alpha = \langle \rho_g \rangle_\alpha \langle T \rangle_\alpha$. If the gas is not isothermal, it is most likely that $\delta\rho_{gi}$ and δT_i will be correlated, since they will both vary proportionally with the inverse of the distance along any radial line (θ and ϕ held constant). In this case, $\langle \rho_g T \rangle_\alpha > \langle \rho_g \rangle_\alpha \langle T \rangle_\alpha$. That is to say, the use of the radial profiles of ρ_g and T will *underestimate* the pressure in regions that are not isothermal.

To find $\langle \rho_g T \rangle_V$, it would be best to use $\langle \rho_g \rangle_V$ and $\langle T \rangle_V$. However, from the radial profiles we have only $\langle \rho_g \rangle_V$ and $\langle T \rangle_M$. It is always the case that $\langle \rho_g \rangle_V < \langle \rho_g \rangle_M$. For the temperature, if T increases as ρ_g increases, then $\langle T \rangle_V < \langle T \rangle_M$. The opposite inequality holds for the converse case in which T decreases as ρ_g increases. In the isothermal case, $\langle T \rangle_V = \langle T \rangle_M$. Again, it is expected that T and ρ_g will both decrease with increasing radius, implying that if the isobars are ellipsoidal, then $\langle T \rangle_V < \langle T \rangle_M$. By this argument, the use of radial profiles which have volume-weighted bins for the gas density and mass-weighted bins for the temperature bins should *overestimate* the pressure in regions that are not isothermal.

It is not expected that these two sources of error would cancel each other. Indeed, since it is found that the clusters studied in this thesis tended to be isothermal in the inner $0.6R_{200}$ and have a shallow dependency with radius compared with the gas density (see Sec. 8.4), the dominant source of error is expected to be the approximation of $\langle \rho_g T \rangle_\alpha$ by $\langle \rho_g \rangle_\alpha \langle T \rangle_\alpha$. However, the analysis demonstrates that in the isothermal region, the mean pressure of the gas can be found exactly in a spherical shell even if the potential well is ellipsoidal in shape. Outside this region, deviations can be expected if the gas distribution is ellipsoidal with the calculated pressure an underestimate of the actual gas pressure.

6.2 A Spherically Symmetric Collapse

How much of an error is introduced by the approximation, $\langle \rho_g T \rangle = \langle \rho_g \rangle \langle T \rangle$? Let us examine a system that is approximately spherically symmetric. A top-hat collapse of gas particles was set up in the following manner. A box of size 15.4 Mpc was filled with 32^3 particles in a regularly spaced manner. All particles had the same mass of $1.3 \times 10^{10}\,\mathrm{M}_\odot = 2.6 \times 10^{43}\,\mathrm{g}$, leading to a mean density of $7.6 \times 10^{-30} \frac{\mathrm{g}}{\mathrm{cm}^3}$. Density perturbations with a spectral index $n = -1$ were created in the regular field by displacement of the positions using the Zel'dovich approximation. Velocities

were assigned to the particles equating to the kinetic energy gained during their displacement into the density perturbations. Those particles within 7.7 Mpc of the box centre were compressed radially by a factor of 0.9 to create a spherical overdensity. This spherical overdensity will ultimately collapse into one object but not until after the smaller-scale features created by the initial density perturbations have collapsed. Thus, the structure formation is a hybrid top-hat-hierarchical process. Those same particles were changed to collisional particles while the particles beyond 7.7 Mpc were changed to dark matter. The initial number of particles participating in the top-hat collapse was then 17156. The contents of the box were evolved for 3×10^9 yrs. The box did not expand (*i.e.*, there was no Hubble flow). The gravitational softening length was 1.9×10^{23} cm $= 59$ kpc and the minimum SPH softening length was half this value.

After the 3×10^9 yrs iteration, the radial profiles of the density and temperature of the cluster were calculated as per Sec. 5.2. From these were calculated radial profiles of the mean gas pressure estimate using $\langle \rho_g \rangle \langle T \rangle$ and the expected gas pressure from the assumption of hydrostatic equilibrium, P_g,

$$P_g = \int \langle \rho_g \rangle (r) \frac{\mathrm{G}\, M(r)}{r^2} dr \tag{6.4}$$

where $M(r)$ is the total mass interior to the shell of radius r. The two curves are shown in Fig. 6.1. There is good correspondence between the estimated mean pressure calculated using Eq. 6.2 and the expected mean pressure using Eq. 6.4. Either the errors in both curves conspire to produce the same offset, the temperature and density are uncorrelated, or the assumption of spherical symmetry is appropriate for this cluster. Since the tophat was initially approximately spherically symmetric, this is the likely answer.

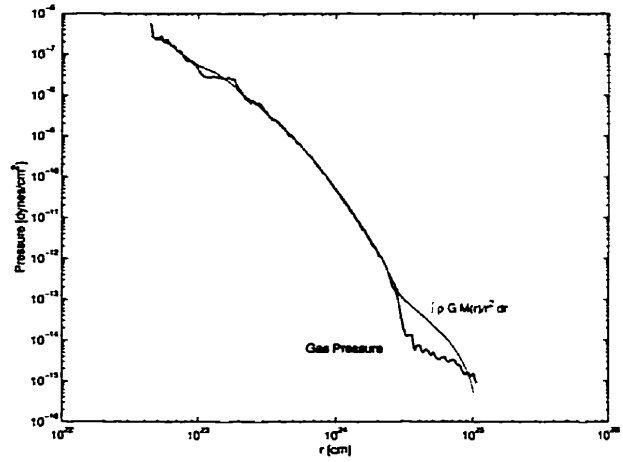


Figure 6.1: The approximate mean gas pressure profile (blue) compared to that expected from the assumption of hydrostatic equilibrium for the cluster formed from a toy top-hat collapse of approximately 17000 gas particles. The outer edge of the initial top-hat radius is 2.4×10^{24} cm. The gravitational softening length is 1.9×10^{23} cm and the minimum SPH softening length is half that. Between these lower bounds and the upper bound of the top-hat radius, the pressure spans approximately 5 orders of magnitude over a decade of radii. Over this span, the two curves agree remarkably well.

6.3 The Hydrostatic State of the Cosmological Clusters

The analysis described above was performed on the sample of cosmological clusters described in Chapter 4. The results for the hierarchically formed clusters, shown in Figures 6.2 and 6.3, indicate that the clusters have mean radial pressure profiles that are consistent with the condition of hydrostatic equilibrium. This is particularly true within the virial radius indicated by R_{200}.

Another indication of this can be gleaned from the ratio of the gas pressure to that pressure expected for the equilibrium situation. Essentially, this is the ratio of the blue curve to the red curve in the vicinity of the green line indicating the overdensity radius R_{200} in Figures 6.2, and 6.3. The distribution of these ratios is given in Fig. 6.4. The distribution indicates that the gas pressure is within a factor of two of the expectation. Interior to this ratio, the profile figures indicate that agreement will generally be even better. Note that values covering a short span in radius are used for calculating this ratio to reduce the noise.

The trend is for the more massive clusters to have a ratio of less than unity (Fig. 6.5) consistent with the halos diverging from spherical symmetry towards ellipsoidal. Only those clusters with masses less than about 5×10^{13} $M_\odot$ have a gas pressure appreciably greater than that expected in the hydrostatic case. Another explanation is that larger clusters are undergoing mergers with smaller objects. This substructure has kinetic energy which supports the gas within the substructure, reducing the necessary gas pressure and consequently reducing the ratio. The outer halos of the low-mass clusters are overheated when in the presence of larger structures.

For the non-hierarchically formed clusters, the gas pressure of the clusters is generally less than that expected if the gas had reached a hydrostatic state (Fig. 6.6) with only three of the seven clusters showing good agreement interior to R_{200} and none

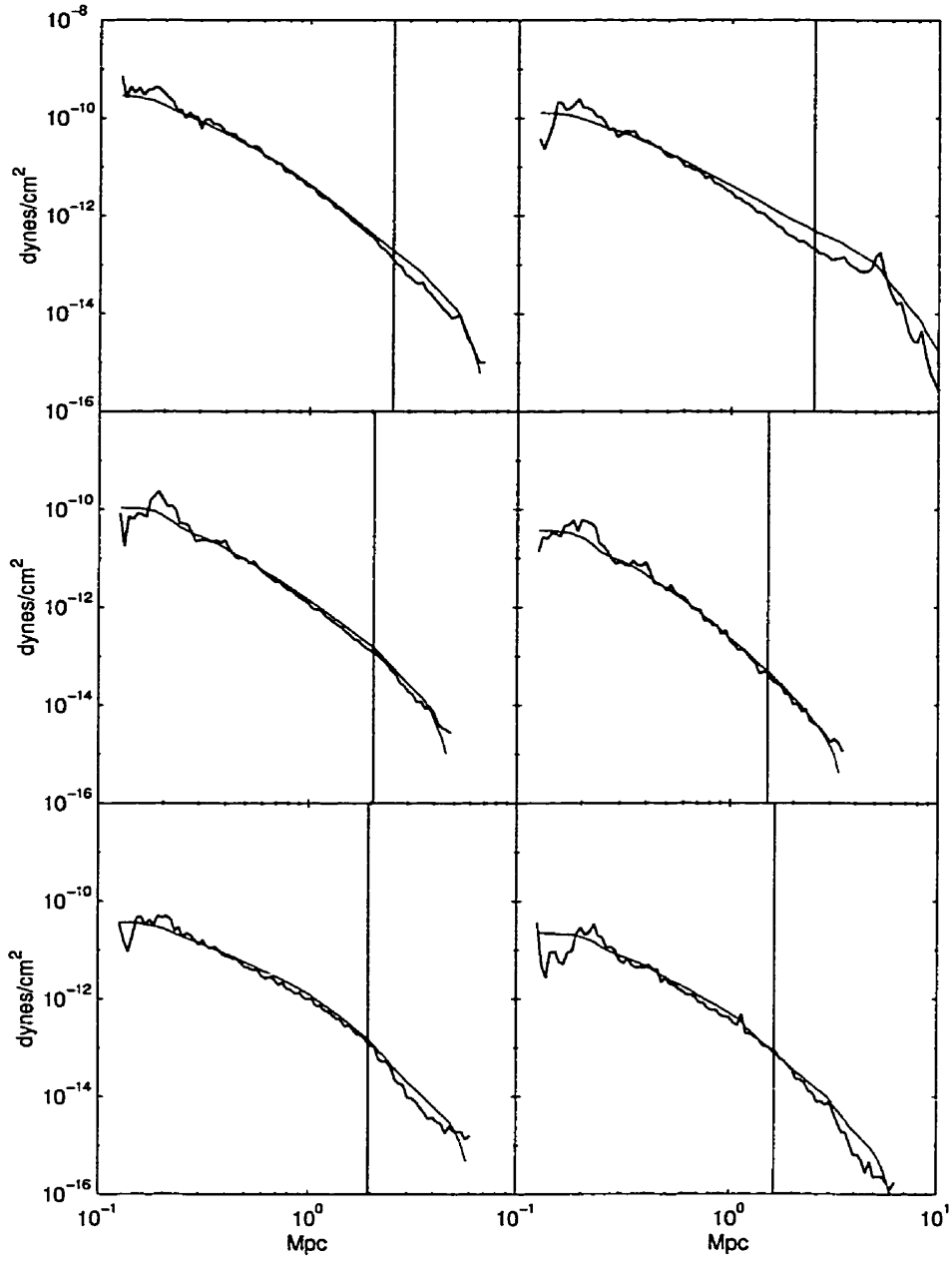


Figure 6.2: Comparison of the gas pressure to the gas pressure expected for a hydrostatic scenario for the hierarchically formed clusters. Plotted are the profiles for the clusters with the six highest central pressures (blue). The red line is calculated from the integration of the equation of hydrostatic equilibrium. The green vertical line denotes R_{200}.

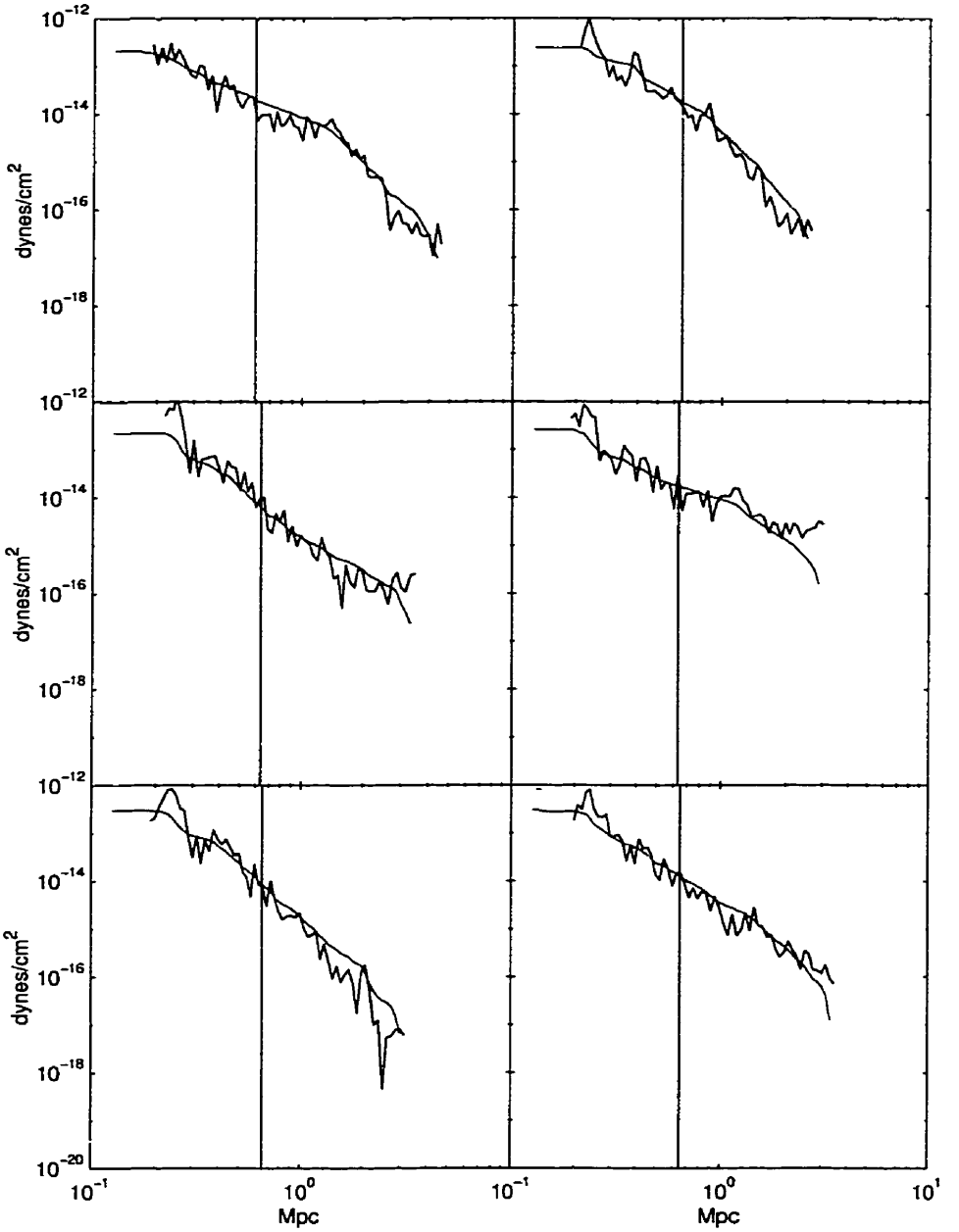


Figure 6.3: Comparison of the gas pressure to the gas pressure expected for a hydrostatic scenario for the hierarchically formed clusters with the lowest central pressures. Compare with Fig. 6.2.

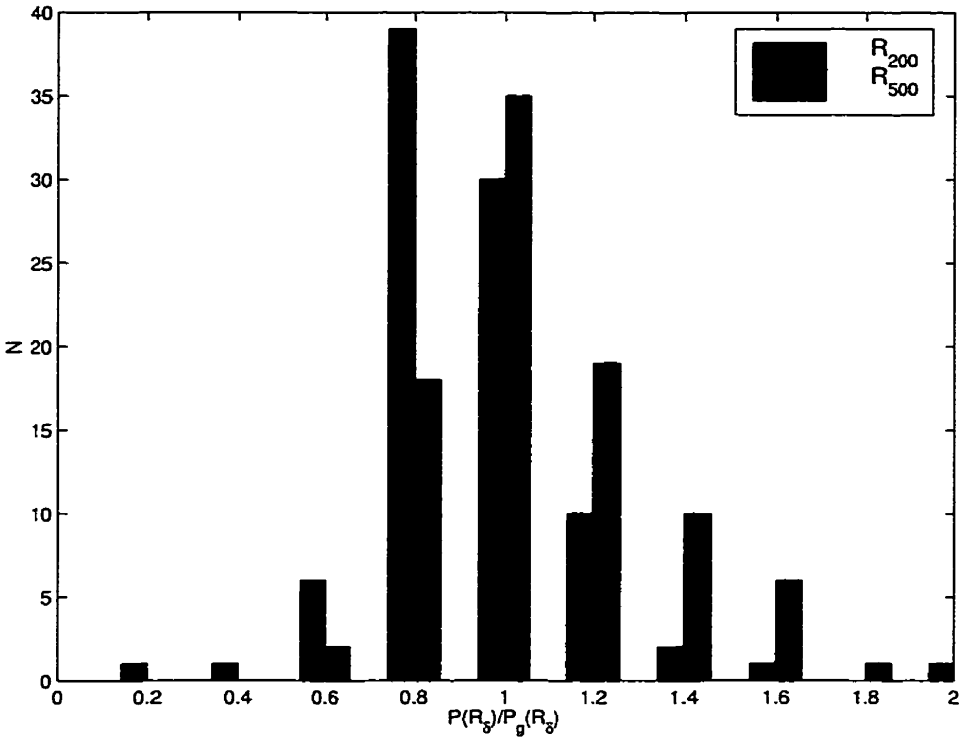


Figure 6.4: The distribution of the ratios of the gas pressure to the pressure expected from the integration of the equation of hydrostatic equilibrium for the hierarchically formed clusters. Given are the ratios taken around the radii R_{200} and R_{500}.

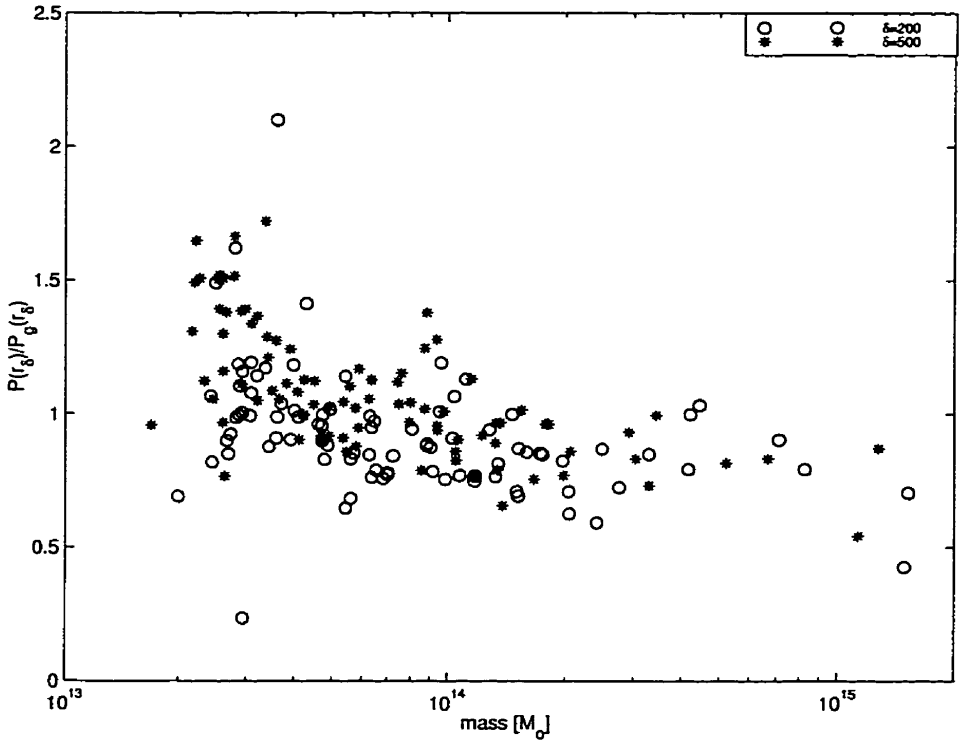


Figure 6.5: The ratios of the gas pressure to the pressure expected from the integration of the equation of hydrostatic equilibrium, plotted versus the cluster mass for the clusters formed hierarchically. Data for ratios at R_{200} and R_{500} and the masses internal to these radii are given.

of them showing good agreement at R_{200}. The ratio of the gas pressures (Fig. 6.7) supports this, particularly in showing that for the smaller radius, R_{500}, the agreement is much better. The ratio being less than unity for all clusters is consistent with the trend for the hierarchically formed clusters since all the clusters are high-mass clusters. However, since there is no substructure in these runs, the ellipsoidal shapes of the clusters must be the cause of the divergence of the calculated and mean pressure profiles.

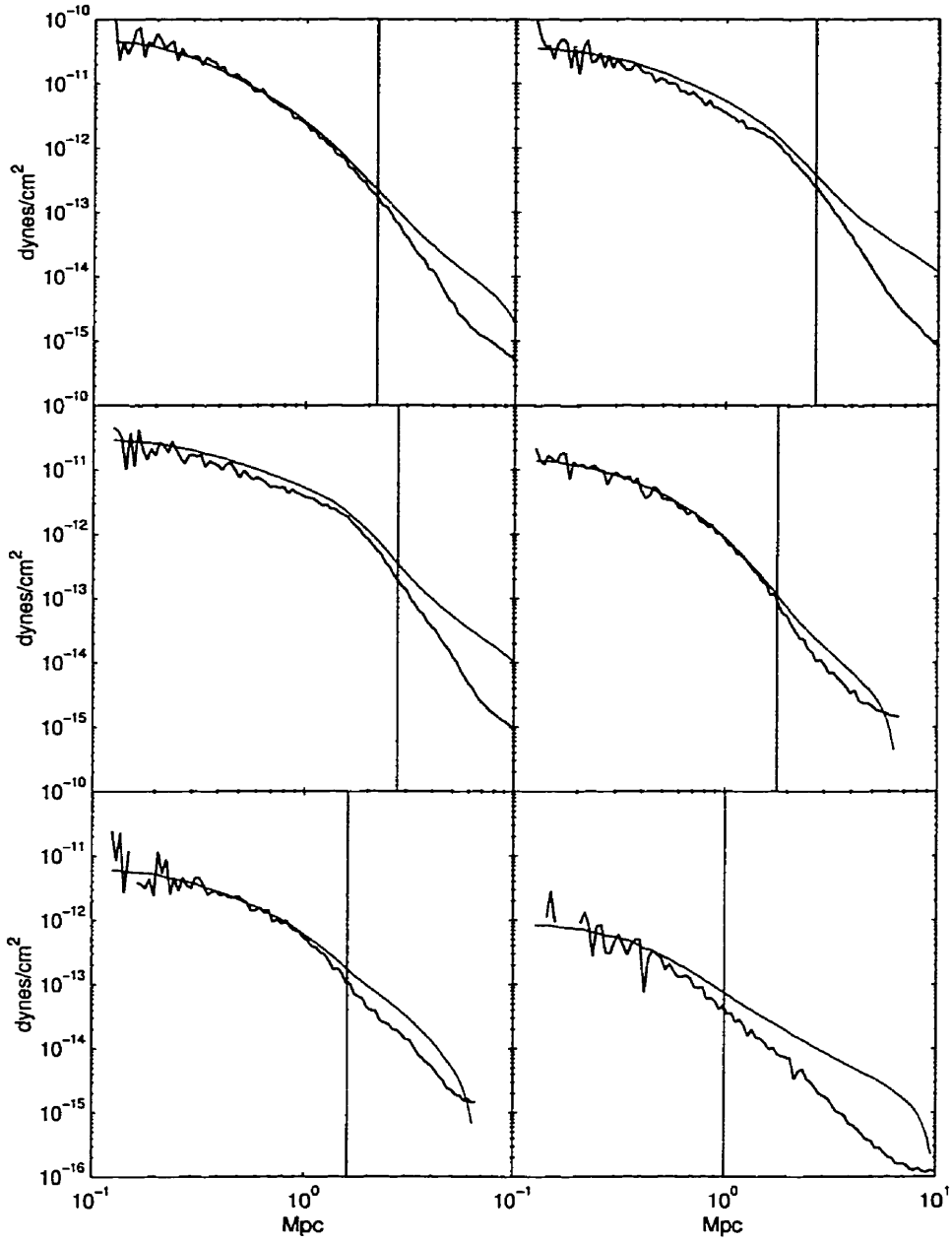


Figure 6.6: Comparison of the gas pressure to the gas pressure expected for a hydrostatic scenario for the non-hierarchically formed clusters. Plotted are the profiles for six of the seven clusters in the sample (blue). The red line is calculated from the integration of the equation of hydrostatic equilibrium. The green vertical line denotes R_{200}.

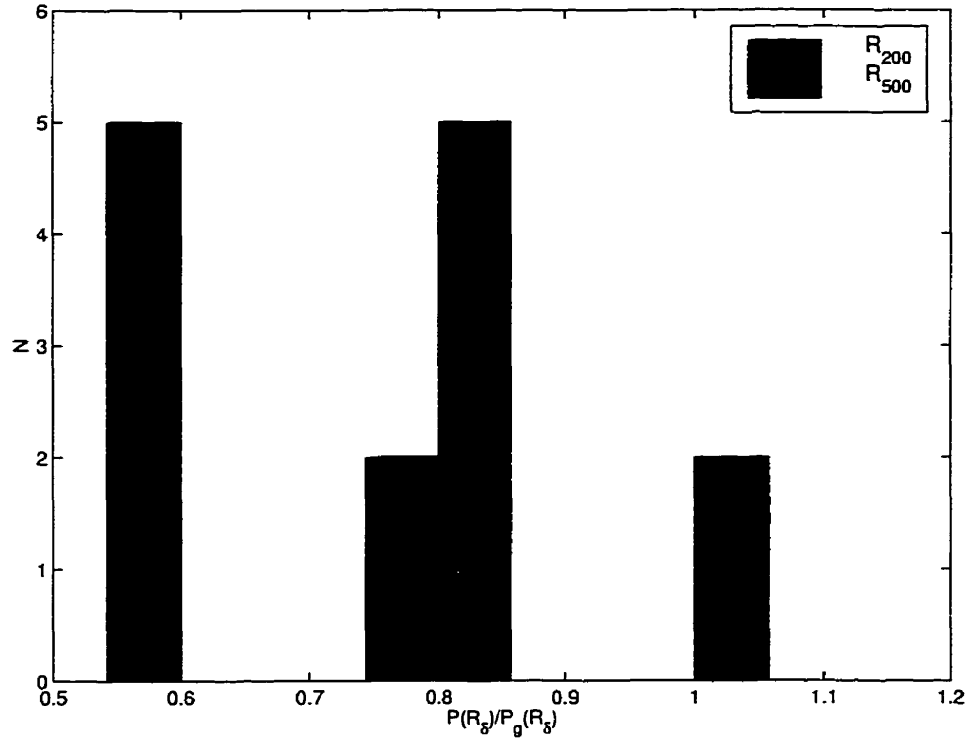


Figure 6.7: The distribution of the ratios of the gas pressure to the pressure expected from the integration of the equation of hydrostatic equilibrium for the non-hierarchically formed clusters. Given are the ratios taken around the radii R_{200} and R_{500}.

Chapter 7

UNIVERSAL PROFILE

Numerical simulations have indicated that, for a Cold Dark Matter (CDM) model, the dark matter density profile of galaxy clusters may be fit by a form with only one free parameter set by the mean density of the universe at the time of collapse (Navarro, Frenk, and White 1996). The existence of such a universal density profile (or NFW profile) would be a useful tool as well as a test for theories of cluster formation. The applicability of the universal profile to clusters formed in other cosmological models would strengthen its use as a tool for observationalists as well as clarify its utililty as a test. Expectations for the shape of the density profile have been made which can explain the existence of a universal profile and support the NFW form (Evans and Collett 1997; Padmanabhan et al. 1996). Some of these depend on hierarchical clustering explicitly.

The dependency of a universal density profile on hierarchical clustering is explored in this chapter. First, the possible forms of the density profile will be examined. After this, the results of fitting the density profiles to clusters formed hierarchically and non-hierarchically will be given.

7.1 Introduction

Over the years, a number of forms for the density profile of a system of collisionless particles has been suggested. Of particular relevance to the dark matter in galaxy clusters are the Hernquist profile and the Navarro, Frenk, and White (NFW) profile. Both have a limiting form in the near and far field with a smooth transition about some characteristic radius.

The Hernquist profile (Hernquist 1990), suggested from observations of spherical galaxies, is given by,

$$\frac{\rho(r)}{\rho_c} = \delta \frac{r_s^4}{r(r_s + r)^3}, \tag{7.1}$$

characterised by a length scale, r_s, and an overdensity, δ.

The work of Navarro, Frenk, and White (1995, 1996, 1997) has built a case for a universal density profile of the similar form,

$$\frac{\rho(r)}{\rho_c} = \delta \frac{r_s^3}{r(r_s + r)^2}, \tag{7.2}$$

Syer and White (1998) (hereafter SW) claims that this form is a byproduct of hierarchical structure formation. They suggest a density profile of the form,

$$\frac{\rho(r)}{\rho_c} = \delta \frac{r_s^\beta}{r^\alpha (r_s + r)^{\beta - \alpha}}. \tag{7.3}$$

The exponents, α and β, correspond to the exponential dependence of the density on radius in the near and far field. That is, for $r \ll r_s, \rho \propto r^{-\alpha}$ and equivalently, for $r \gg r_s, \rho \propto r^{-\beta}$.

The forms of Hernquist and NFW are particular cases of the SW density profile. For the Hernquist profile, $\alpha = 1$ and $\beta = 4$. Correspondingly for the NFW profile, $\alpha = 1$ and $\beta = 3$. It is the shape of the density profile in the inner radii which

determines α. The inner radii, however, are affected strongly by the resolution of the simulation. Indeed, the cluster profiles generally span only two orders of magnitude in radius. Any attempt to fit a smoothly varying curve over this span and then glean information about the near-field and far-field dependency of the profile is thus problematic.

In order to circumvent the ambiguity in derived values of α and β due to the limited range of the fit, a non-smoothly varying form is required which is discontinuous in the first derivative at the 'knee' separating the nominal near and far fields. This suggests the following functional form should be fit,

$$\frac{\rho(r)}{\rho_c} = \begin{cases} \delta_\alpha r^{-\alpha}, & r < r_s; \\ \delta_\beta r^{-\beta}, & r > r_s; \end{cases} \tag{7.4}$$

Since the profile must be continuous at r_s,

$$\delta_\beta = \frac{r_s^\beta}{r_s^\alpha}\delta_\alpha. \tag{7.5}$$

There are several predictions for the shape of the density cusp. They usually involve discussions of the growth of structure from a self-similar initial density distribution. An exception to this is the result of Evans and Collett (1997). Looking at the stability of clusters to perturbations due to binary encounters, they showwed that $\alpha = 4/3$ is a stable solution of both the Fokker-Planck and collisionless Boltzmann equations.

Hoffman and Shaham (1985) assumes a spherically symmetric halo collapse to derive a radial dependency for the dark matter density of the form

$$\rho(r) \propto r^{-\frac{3(n+3)}{(n+4)}} \tag{7.6}$$

for the inner cusp and $\rho(r) \propto r^{-4}$ in the outer halo. A similar result is found using a more detailed analysis in Padmanabhan et al. (1996). Using the assumptions of

self-similarity and stability in the form of the cluster in the non-linear regime (that is, the morphology of a virialised object only scales with time), Padmanabhan et al. (1996) derives a relation for the 2-point correlation function in the non-linear stage of evolution from an initial linear density perturbation with a spectral index of n,

$$\xi(r) \propto r^{-\frac{3(n+3)}{(n+5)}}. \tag{7.7}$$

Since the 2-point correlation function is simply the excess probability of finding any particle at a distance r from a given particle, if most of the matter is already located in the high-density regions, it is approximately related to the density profile by

$$\frac{\rho(r)}{\rho_{bg}} \simeq 1 + \xi(r). \tag{7.8}$$

This predicts $\alpha = 3(n+5)/n + 5$ for our density profiles. The same result is derived in a slightly different form in Padmanabhan (1996). Linear theory for the growth of clusters from an initial density fluctuation power spectrum, $P(k) \propto k^n$ predicts for $\rho \propto r^\alpha$ in the case of $r \ll r_s$.

$$\alpha = 3\left(\frac{3+n}{5+n}\right). \tag{7.9}$$

Extending the previous arguments to hierarchical clustering, Syer and White (1998) argues that the initial cusp form found in the first objects formed will be maintained despite mergers. Consider a cluster absorbing a smaller satellite. If the density cusp in the cluster is steeper than that of the satellite, then the satellite will be tidally destroyed, softening the cluster cusp. If the satellite cusp is steeper, it will survive tidal disruption and sink to the bottom of the cluster, steepening the cluster cusp.

Based on the self-similar infall model, which excludes hierarchical clustering, Henriksen and Widrow (1998) finds $\rho \propto r^{-2}$ for the inner dark matter density profile and r^{-3} in the outer limits.

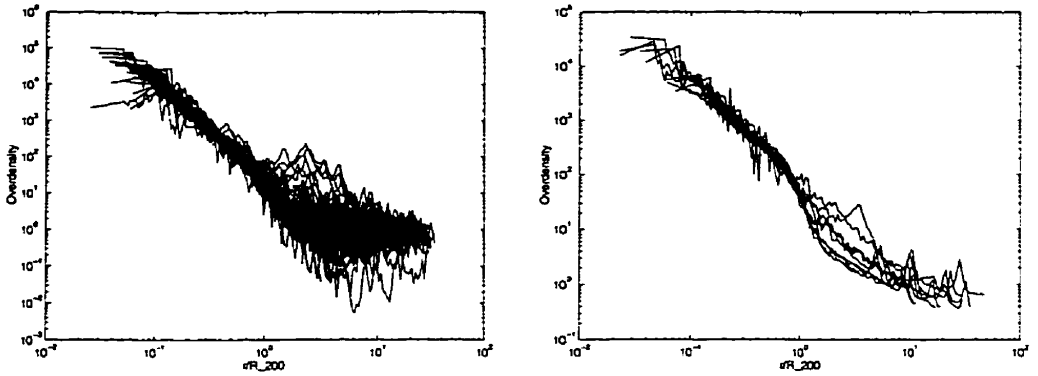


Figure 7.1: Dark matter density profiles in the unsmoothed run (left), and smoothed run (right) at the time corresponding to the present. The radii have been scaled to R_{200}.

7.2 The Density Profiles of The Simulated Clusters

The density profile for the dark matter scales well between clusters in the run in which substructure was present (Fig. 7.1, left). When the radii are scaled to R_{200} for each cluster, it is clear that there are three regimes, nominally the near-field, the far-field, and the background, each with distinctly different power law dependencies.

The four forms of density profiles (NFW, Hernquist, SW, and the form suggested here, Eq. 7.4) were fit to the scaled SPH-estimated densities of the particles for the unsmoothed hierarchical run (Fig. 7.2). The method of fitting minimises χ^2. To separate the mean background density from the fitting, only the span $r < 2R_{200}$ was fit.

All forms fit the data equally well. However, due to the limited range in radii over which to fit the profiles (typically only two orders of magnitude), the limiting cases of the near and far field are poorly discerned. In particular, the value for α is poorly constrained. It is apparent that the 'knee' position, r_s, and either the near-field dependency, α, in the hierarchical case or the far-field dependency, β, are being used as free parameters to shape the smooth transition between the two regimes. The

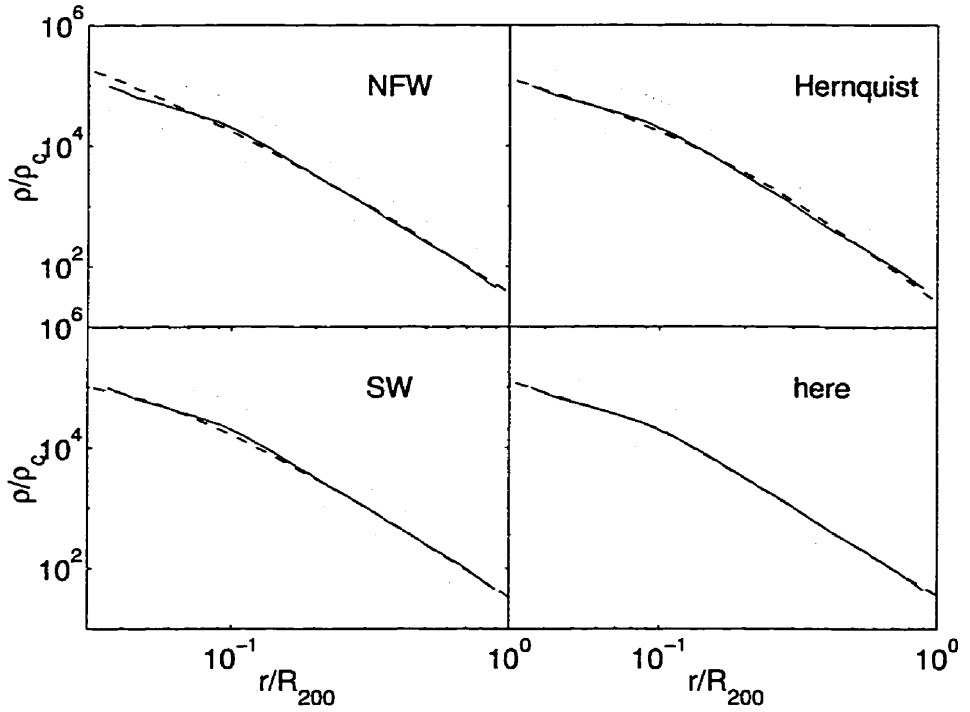


Figure 7.2: Hierarchical data. Fits to the mean dark matter density profile for the forms of Navarro, Frenk, and White (top left), Hernquist (top right), Syer and White (lower left), and the form suggested here (lower right). The dashed lines are the fits. The solid lines are the mean profiles with 1σ variations bound by the dotted lines.

	r_s	α	β	χ^2/N
NFW	0.06 ± 0.01	1	3	0.98
Hernquist	0.19 ± 0.01	1	4	1.00
SW	0.0008 ± 0.003	-7 ± 3	2.94 ± 0.02	0.98
here	0.12 ± 0.01	1.84 ± 0.03	2.74 ± 0.01	0.97

Table 7.1: Coefficients for fits to the dark matter density profiles for the hierarchical case using a variety of profile forms. The SW fit, though free to fit an inner slope, fails to converge to a value for α and fits, instead, a single power law.

SW fit fails entirely to discern an inner slope. However, for the outer slope, the SW fit as well as the form suggested here agree with the expectation of NFW.

For these simulations, $n = -1$ which gives $\alpha = 1.5$ using the predictions of Eq. 7.7 and Eq. 7.9. This compares quite well with the results of the fit to the form suggested and agrees with the results of fitting to the form suggested by Syer and White to the non-hierarchical data, for which the near-field is properly fit. However, there is no reason that Eq. 7.9 should hold for non-hierarchical growth.

Other groups find similar results. For a Cold Dark Matter (CDM) initial power spectrum, Moore et al. (1997) and Moore et al. (1998) find $\alpha = 1.4$ in a high-resolution (softening of 5 kpc) collisionless simulation. More generally, Fukushige and Makino (1997) finds for the core cusp $\alpha = 1$ to 2, in high-resolution collisionless simulations of a CDM model using the GRAPE special-purpose N-body hardware. The authors reported a resolution of 1 kpc in these simulations.

The results of the fits for all the models (Table 7.2) indicate that the NFW fits do as well as the less constrained fit suggested here, even in the non-hierarchical cases. The scale radius increases with increased smoothing for the NFW fit in the high resolution simulation, but not the low resolution one. The inner profiles, as parametrised by α, have no dependency on the degree of smoothing but do exhibit a great deal of scatter. The mean value for the high resolution runs is $\langle\alpha\rangle = 1.8 \pm 0.2$. The outer profile does tend to become less shallow as the smoothing increases. This is qualitatively consistent with Crone, Evrard, and Richstone (1994) who find the profile becomes more shallow as the initial power spectrum index, n, is increased in a set of pure N-body simulations of clusters.

Similar to the situation being tested here, Huss, Jain, and Steinmetz (1998) have performed a set of N-body only simulations of an isolated cluster in which the amount

Run	Form	r_s	α	β	χ^2/N
unsmoothed	NFW	0.06 ± 0.01	1	3	0.98
	here	0.12 ± 0.01	1.84 ± 0.03	2.74 ± 0.01	0.97
3 Mpc	NFW	0.10 ± 0.01	1	3	0.98
top-hat	here	0.07 ± 0.01	1.5 ± 0.2	2.46 ± 0.01	0.98
$(7\,\mathrm{Mpc})^{-1}$	NFW	0.09 ± 0.01	1	3	0.97
k-cutoff	here	0.3 ± 0.2	2.1 ± 0.2	2.8 ± 0.2	0.99
7 Mpc	NFW	0.28 ± 0.01	1	3	0.99
top-hat	here	0.23 ± 0.01	1.70 ± 0.03	2.23 ± 0.01	0.98
$(14\,\mathrm{Mpc})^{-1}$	NFW	0.21 ± 0.01	1	3	0.99
k-cutoff	here	0.18 ± 0.01	1.78 ± 0.03	2.28 ± 0.01	0.98
low-res.	NFW	0.26 ± 0.03	1	3	1.01
unsmoothed	here	0.23 ± 0.01	2.08 ± 0.06	2.38 ± 0.05	1.01
low-res.	NFW	0.24 ± 0.02	1	3	1.00
7 Mpc	here	0.26 ± 0.1	4 ± 2	2.4 ± 0.1	1.40
top-hat					

Table 7.2: Coefficients for fits to the dark matter density profiles for all the models.

of substructure is controlled via manipulation of the velocity dispersion of the particles. From this data set, they conclude that the NFW form maintains its universality in non-hierarchical scenarios.

Modelling clusters in a variety of cosmologies at very high resolution (256^3 particles) Thomas et al. (1998) found the density profiles of the clusters to follow the form of the NFW profile. These simulations were N-body only.

Chapter 8

MASS-TEMPERATURE SCALING LAW

In a hierarchical clustering scenario, the process of merging 'recreates' the cluster morphology on a continual basis. Clusters are all being formed at the present from structures formed at a variety of ages which are, in turn, formed by the amalgamation of many other smaller structures. In a non-hierarchical clustering scenario, the matter in the clusters has actually collapsed for the first time only recently. In both cases, if hydrostatic equilibrium has been established and there exist density and temperature profiles common to all clusters, then it can be expected that there will be a common mass-temperature scaling law for the clusters in a given clustering scenario.

8.1 Derivation of The Mass-Temperature Scaling Law

The equation of hydrostatic equilibrium relates the pressure, $P(r)$, with the the mass, $M(r)$, internal to the radius, r, via

$$\frac{GM(r)}{r^2} = -\frac{1}{\rho}\frac{dP}{dr}, \tag{8.1}$$

where ρ is the gas density and G retains its normal use as the gravitational constant.

Combined with the perfect gas law,

$$P = \frac{\mathrm{k}\,\rho T}{\mu m_u}, \tag{8.2}$$

which relates the pressure to the temperature, T, with k, μ, and m_u being Boltzmann's constant, the mean molecular weight, and the atomic mass unit, respectively, Eq. 8.1 gives,

$$\frac{GM(r)}{r} = -\frac{kT}{\mu m_u}\left(\frac{r}{\rho}\frac{d\rho}{dr} + \frac{r}{T}\frac{dT}{dr}\right). \tag{8.3}$$

Since we have the definition of ln given by

$$\frac{1}{x} = \frac{d}{dx}\ln(x), \tag{8.4}$$

the following relation is found:

$$\frac{GM(r)}{r} = -\frac{kT(r)}{\mu m_u}\left[\frac{d\ln\rho}{d\ln r} + \frac{d\ln T}{d\ln r}\right]. \tag{8.5}$$

It is shown in Sections 8.3 and 8.4 that both the gas density and temperature profiles can be approximated by power laws away from the 'knee' radius. Take the exponents of these respective profiles to be n_ρ and n_T. This gives,

$$\rho(r) \propto r^{n_\rho}, T(r) \propto r^{n_T} \qquad \Rightarrow \frac{GM(r)}{r} = -\frac{kT(r)}{\mu m_u}\left[n_\rho + n_T\right]. \tag{8.6}$$

In general, n_ρ and n_T are negative.

The properties of clusters are characterised by a radius, $R_{\bar{\delta}}$, and mass, $M_{\bar{\delta}}$, of some overdensity. This overdensity is typically taken to be around the value of the virial overdensity for a top-hat collapse, *i.e.* $\bar{\delta} = 178$. However, the characteristic radius is not particularly sensitive to the choice of $\bar{\delta}$. Given this description, we have the definition

$$R_{\bar{\delta}} = \left[\frac{3}{4\pi}\frac{M_{\bar{\delta}}}{\bar{\delta}\rho_c}\right]^{1/3} \tag{8.7}$$

which, when combined with Eq. 8.6, gives

$$\mathrm{G}\left[\frac{4\pi}{3}\bar{\delta}\rho_c\right]^{1/3} M_{\bar{\delta}}^{2/3} = -\frac{kT(R_{\bar{\delta}})}{\mu m_u}\left[n_\rho + n_T\right]. \tag{8.8}$$

Thus the scaling relationship, $T \propto M^{2/3}$, is found. Recall that the critical density is given by,

$$\rho_c = \frac{3(h100\,\mathrm{km\,s^{-1}\,Mpc^{-1}})^2}{8\pi\,\mathrm{G}} \tag{8.9}$$

with h defined by its relation to the Hubble constant, $H_o \equiv h100\frac{\mathrm{km/s}}{\mathrm{Mpc}}$. On the scales of interest, this relation may be written as,

$$\frac{T(R_{\bar{\delta}})}{1\,\mathrm{keV}} = \frac{2.83h^{2/3}\bar{\delta}^{1/3}}{-(n_\rho + n_T)}\left[\frac{M_{\bar{\delta}}}{10^{15}\,\mathrm{M}_\odot}\right]^{\frac{2}{3}}. \tag{8.10}$$

Significantly, the coefficient of the mass-temperature relation is dependent on the forms of the gas density and temperature profiles through the exponents n_ρ and n_T.

8.2 The Temperature Parameter: τ

Since the temperatures of the clusters are scaled by the mass of the cluster, it is convenient to introduce the parameter

$$\tau(r) \equiv \frac{\frac{T(r)}{1\,\mathrm{keV}}}{\left(\frac{M_{200}}{10^{15}\,\mathrm{M}_\odot}\right)^{\frac{2}{3}}}. \tag{8.11}$$

Thus, the τ profile for a cluster will have the same form as $T(r)$ but will be scaled. Hence, if $T(r) \propto r^{n_T}$ then $\tau(r) \propto r^{n_T}$ as well.

For ease of comparison, two temperature parameters are defined. The first,

$$\tau_{num} \equiv \frac{\frac{T(r=0)}{1\,\mathrm{keV}}}{\left(\frac{M_{200}}{10^{15}\,\mathrm{M}_\odot}\right)^{\frac{2}{3}}}, \tag{8.12}$$

can have its value found from the distribution of temperature versus mass for the clusters.

Given that Eq. 8.10 gives $T(R_{\bar{\delta}})$ while τ_{num} is related to $T(r=0)$, we can compare the two by including a factor for the scaling between $r = R_{\bar{\delta}}$ and $r = R_{iso}$, the radial limit of the isothermal core. This gives for the second temperature parameter,

$$\tau_{analy} = \frac{2.83h^{2/3}\bar{\delta}^{1/3}}{-(n_\rho + n_T)} R_{iso}^{n_T}, \tag{8.13}$$

which will allow us to compare the actual central temperatures with the central temperature to be expected for a cluster with density and temperature radial profiles at $R_{\bar{\delta}}$ which have power-law dependencies of n_ρ and n_T respectively as well as isothermal radii approximated by R_{iso}.

8.3 Mean Gas Density Profiles

The gas densities were recalculated using the SPH density estimator (Eq. 3.4) with the smoothing lengths set to enclose exactly $N_{SPH} = 32$ particles. The distances from the cluster centres, scaled by the overdensity radius R_{200} for each of the clusters, were found for each particle. The sample of particles comprised those that are within $2R_{200}$ but beyond the point interior to which there were 250 particles (see Sec. 3.1). Fig. 8.1 illustrates the sample for the clusters formed hierarchically. The cluster profiles scale remarkably well. Together, these points give a set of data points to which a density profile was fit. The fit used was the discontinuous density profile form given by Eq. 7.4 introduced in Sec. 7.2. The results are summarised in Table 8.1.

There is a weak trend in which r_s increases with increased smoothing. A much more significant trend is for α to decrease with increased smoothing. That is to say the density profile becomes more shallow as smoothing is increased. The outer profile changes little with a power-law dependency on radius of ≈ -2.7 which is slightly shallower than the density profile for the dark matter (see Sec. 7.2).

The standard form to which gas density profiles are fit is the "β"-model (Cavaliere and Fusco-Femiano 1976) which describes the expected density profile of an isothermal

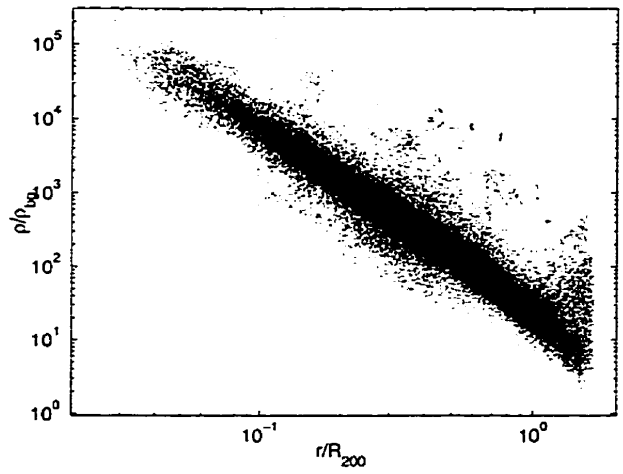


Figure 8.1: Gas densities for the hierarchical run. The densities for the sample of particles described in the text. The radii have been scaled to R_{200}. The total sample corresponds to about 30% of all gas particles.

	r_s	α	β	χ^2/N
unsmoothed	0.43 ± 0.03	2.37 ± 0.01	2.45 ± 0.01	0.96
3 Mpc top-hat	0.56 ± 0.03	2.27 ± 0.01	2.80 ± 0.04	0.93
$(7\,\mathrm{Mpc})^{-1}$ k-cutoff	0.54 ± 0.01	2.24 ± 0.01	2.97 ± 0.02	0.94
7 Mpc top-hat	0.59 ± 0.01	1.60 ± 0.02	2.99 ± 0.03	1.00
$(14\,\mathrm{Mpc})^{-1}$ k-cutoff	0.23 ± 0.01	1.78 ± 0.03	2.34 ± 0.01	0.94
low-res. unsmoothed	0.43 ± 0.05	3 ± 2	2.34 ± 0.03	1.00
low-res. 7 Mpc top-hat	0.73 ± 0.03	1.54 ± 0.08	3.0 ± 0.1	1.01

Table 8.1: Coefficients for fits to the gas density profiles. All use the discontinuous form.

cloud. This model is used frequently in the interpretation of X-ray observations of galaxy clusters (Fabricant and Gorenstein 1983; Jones and Forman 1984). It has the continuous form

$$\rho(r) = \rho_o \left[1 + \left(\frac{r}{r_s}\right)^2\right]^{-\frac{3}{2}\beta}. \tag{8.14}$$

The significance of β in this fit is that it also represents the ratio of the galaxy (or more specifically, collisionless baryonic component) kinetic energy to gas thermal energy. That is,

$$\beta \equiv \frac{\sigma^2}{\mathrm{k}T/\mu m_u} \tag{8.15}$$

where σ is the one dimensional velocity dispersion. Though there is no collisionless baryonic component in these simulations and the gas is well thermalised, it may still be interesting to see if the inferred value of β varies among the models. When Eq. 8.14 is fit to the mean gas profiles (Table 8.2), it is found that β varies little among the models with a mean value of 0.82 ± 0.03 for the high resolution runs. This compares with the value of 0.76 ± 0.06 found by Tsai, Katz, and Bertschinger (1994) who analysed a single cluster simulated at high resolution in a CDM model. Eke, Navarro, and Frenk (1998) simulated clusters in a low-density universe and found a mean value of 0.74 ± 0.15.

8.4 Mean τ Profile

Since the cluster temperatures are scaled by the mass, it can be expected that the temperature profiles are also accordingly scaled. As such, the profile of the scaled-temperature parameter, τ defined in Eq. 8.11, should be used to compare temperature distributions.

A data set for the temperature parameter, τ, was created in a manner similar to that of the gas density data set in Sec. 8.3. For the hierarchical case, the mean

	r_s	β	χ^2/N
unsmoothed	0.020 ± 0.001	0.808 ± 0.001	0.96
3 Mpc top-hat	0.044 ± 0.001	0.840 ± 0.002	0.94
$(7\,\mathrm{Mpc})^{-1}$ k-cutoff	0.056 ± 0.001	0.868 ± 0.002	0.95
7 Mpc top-hat	0.145 ± 0.005	0.811 ± 0.001	1.04
$(14\,\mathrm{Mpc})^{-1}$ k-cutoff	0.073 ± 0.002	0.791 ± 0.004	0.94
low-res. unsmoothed	0.02 ± 0.02	0.78 ± 0.02	4.05
low-res. 7 Mpc top-hat	0.42 ± 0.04	1.05 ± 0.05	1.02

Table 8.2: Coefficients for the β-fit to the gas density profiles.

scaled τ profile is shown in Fig. 8.2. There is a great deal more scatter than for the density. Some of this is due to smaller clusters being satellites of larger clusters. The halo gas of these smaller clusters gets shocked to high temperatures which then are scaled by the mass of the smaller cluster. However, there is an approximately isothermal core that extends to $\approx 0.2R_{200}$. The outer profile may be crudely fit by a power-law dependency on radius. Fits to the profiles with the discontinuous form given in Eq. 7.4 which has a free power-law index for the inner regime, α, find values of α spanning -0.05 to 0.10. For this reason, an approximate τ-profile is fit using a form similar to the discontinuous form but with an iso-thermal core:

$$\tau(r) = \begin{cases} \tau_o, & \frac{r}{R_{200}} < R_{iso}; \\ \tau_o \left(\frac{R_{200}}{R_{iso}}\right)^{n_T} \left(\frac{r}{R_{200}}\right)^{n_T}, & \frac{r}{R_{200}} > R_{iso}; \end{cases} \tag{8.16}$$

where τ_o is the scaled temperature (in units of $T/M^{2/3}$) in the centre of the cluster. The results of fitting this to all the temperature profiles are given in Table 8.3.

The central temperatures are found to generally decrease with increased smoothing. However, the trend is not clear. The isothermal core radius increases slightly with increasing smoothing, in step with the trend found for r_s of the gas-density profiles, but marginally interior to r_s. The power-law dependency on radius, n_T, shows a trend to steepen slightly with increased smoothing, ranging from -0.4 to -0.6. This

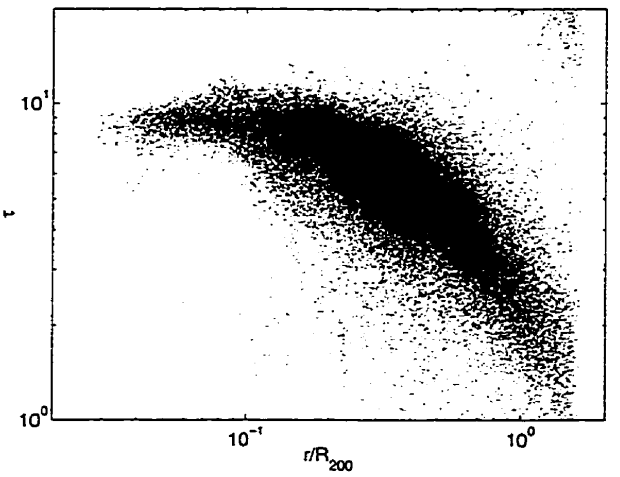


Figure 8.2: The scaled-temperature parameter, τ, for the hierarchical run. The sample of particles is the same as in Fig. 8.1.

	τ_o	R_{iso}	$-n_T$	χ^2/N
unsmoothed	8.5 ± 1	0.10 ± 0.03	0.41 ± 0.04	1.02
3 Mpc top-hat	8.6 ± 0.1	0.10 ± 0.03	0.47 ± 0.01	1.01
$(7\,\mathrm{Mpc})^{-1}$ k-cutoff	7.52 ± 0.04	0.16 ± 0.01	0.57 ± 0.01	1.03
7 Mpc top-hat	4.85 ± 0.04	0.26 ± 0.01	0.56 ± 0.02	1.01
$(14\,\mathrm{Mpc})^{-1}$ k-cutoff	6.59 ± 0.04	0.18 ± 0.01	0.57 ± 0.01	1.06
low-res. unsmoothed	9.9 ± 0.5	0.5 ± 0.1	0.7 ± 0.2	1.03
low-res. 7 Mpc top-hat	4.0 ± 0.1	0.58 ± 0.02	0.92 ± 0.07	1.00

Table 8.3: Coefficients for fits to the gas temperature profiles. All use a discontinuous form with an iso-thermal core.

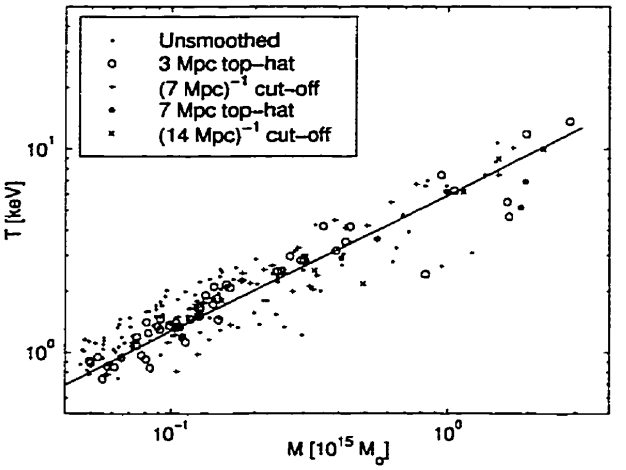


Figure 8.3: The temperature *vs* mass for the clusters. The solid line is the fit $T = 5.9M^{2/3}$.

value has a strong dependency on resolution, with $n_T \approx -0.8$ for the lower resolution runs. As such, the profile may become even more shallow with a further increase in resolution. This may be due to a lack of shock-heating due to the lower particle numbers in the outer halos.

8.5 Mass-Temperature Distribution

Temperatures within $0.2R_{200}$ are essentially constant (see Sec. 8.4). This distance was used as the cutoff radius in determining the mean cluster temperature, T_r. The distribution of temperature versus mass is illustrated in Fig. 8.3.

The correlation between central cluster temperature, T_r, and the mass of the cluster is found to obey the relationship,

$$T_r = \tau_{num} \left(\frac{M_{200}}{10^{15}\,\mathrm{M}_\odot} \right)^{2/3} \mathrm{keV}, \tag{8.17}$$

where τ_{num}, the coefficient peculiar to a run, is given in Table 8.4. The data points were assumed to be independent for the sake of the fit and error estimate. That is,

	τ_{num}
unsmoothed	6.8 ± 1.6
3 Mpc top-hat	6.2 ± 1.1
$(7\,\mathrm{Mpc})^{-1}$ k-cutoff	6.2 ± 1.1
7 Mpc top-hat	4.9 ± 0.9
$(14\,\mathrm{Mpc})^{-1}$ k-cutoff	5.5 ± 1.1
low-res. unsmoothed	5.8 ± 1.3
low-res. 7 Mpc top-hat	4.9 ± 1.0

Table 8.4: Mass-temperature scaling law coefficient. The coefficient, A, of Equation 8.17 is given for the simulations. It relates the cluster temperature (in keV) with the cluster mass (in $10^{15}\,\mathrm{M}_\odot$).

the error is,

$$\sigma = std\left[\frac{T_r/\,\mathrm{keV}}{\left(\frac{M_{200}}{10^{15}\,\mathrm{M}_\odot}\right)^{2/3}}\right]. \tag{8.18}$$

There may be a weak (1σ) dependence of τ_{num} on the degree of smoothing in the initial conditions. The value of τ_{num} drops from 7 ± 1.6 in the unsmoothed case to 5 ± 1 for the 7 Mpc smoothing. This is consistent with the trend in τ_o found for the fits to τ-profile.

If the exponent, $n = 2/3$, in Eq. 8.17 is allowed to be free, the dependency on $M^{2/3}$ is confirmed (Table 8.5). The free exponent, $n = 0.67 \pm 0.04$, is consistent with $n = 2/3$. Interestingly, the dependency of τ_{num} on smoothing seems to disappear in this case.

For Table 8.5, errors were estimated using a bootstrap approach. Fits were calculated repeatedly using cluster data from a sample of half of the clusters, chosen randomly. The errors were determined from the variance of the coefficients of these fits. This method, of course, biases towards the more common small-mass clusters whose counterparts in the real world would be less readily observed. When the samples are restricted to the more massive clusters, the results do not change significantly. Fits

	τ_{num}	n
unsmoothed	5.57 ± 0.50	0.61 ± 0.03
3 Mpc top-hat	5.89 ± 0.61	0.67 ± 0.04
$(7\,\mathrm{Mpc})^{-1}$ freq. cut	6.96 ± 0.63	0.75 ± 0.04
7 Mpc top-hat	4.80 ± 0.86	0.67 ± 0.12
$(14\,\mathrm{Mpc})^{-1}$ freq. cut	6.96 ± 0.63	0.75 ± 0.04
low-res. unsmoothed	5.7 ± 0.5	0.66 ± 0.07
low-res 7 Mpc top-hat	5.7 ± 1.5	0.68 ± 0.26

Table 8.5: Mass-temperature scaling law with free coefficients.

to the data points assume a correlation among clusters and, as such, lead to smaller errors than are found assuming independence among clusters (cf. Eq. 8.18).

The value for the coefficient, τ_{num}, is in agreement with the results of Navarro, Frenk, and White (1995), and Evrard, Metzler, and Navarro (1996) which report values of 5.11 and 5.20 for τ_{analy}. Note that for this last value, the empirical translation $M_{200} \approx 1.2 M_{500}$ has been assumed here. From the data presented here, this ratio is found to be 1.17 ± 0.11 and a slight trend towards higher values for more massive clusters is noted. For the previous assumption, this trend is insignificant. Using a Eularian code, Bryan and Norman (1998) found a lower value of 4.7 ± 0.1 from a variety of CDM models. Various authors have found observational evidence supporting this scaling law (Schindler 1996; Tsai, Katz, and Bertschinger 1994) for which they find $T = 7.8\,\mathrm{keV}(M_{Tot}/10^{15}\,\mathrm{M}_{\odot})$. Recently, Balogh, Babul, and Patton (1998) has shown that if the gas is preheated and allowed to collapse adiabatically in to an isothermal potential well, then the mass-temperature scaling relationship overestimates the halo masses by up to an order of magnitude. This occurs for halos the size of groups of galaxies or less ($M < 10^{14}\,\mathrm{M}_{\odot}$), which is at the low end of the size of the halos examined here. Otherwise, it recovers the relationship found here, with a coefficient of about 4.5.

	τ_{analy}	τ_{num}	τ_o
unsmoothed	7.7 ± 0.2	6.8 ± 1.6	8.5 ± 1
3 Mpc top-hat	6.8 ± 0.2	6.2 ± 1.1	8.6 ± 0.1
$(7\,\mathrm{Mpc})^{-1}$ k-cutoff	7.1 ± 0.3	6.2 ± 1.1	7.52 ± 0.04
7 Mpc top-hat	6.4 ± 0.7	4.9 ± 0.9	4.85 ± 0.04
$(14\,\mathrm{Mpc})^{-1}$ k-cutoff	7.1 ± 0.3	5.5 ± 1.1	6.59 ± 0.04
low-res. unsmoothed	7.6 ± 0.4	5.8 ± 1.3	9.9 ± 0.5
low-res. 7 Mpc top-hat	5.4 ± 0.4	4.9 ± 1.0	4.0 ± 0.1

Table 8.6: Analytic and numeric coefficients for the M-T relation. Given are the semi-analytic estimates for the scaled temperature parameter, τ, the numerical value from fitting to the temperature-mass distribution, and the values derived from fitting the mean profiles.

8.6 Comparison With The Semi-analytic Prediction

Recall Eq. 8.13 defines,

$$\tau_{analy} \equiv \frac{2.83 h^{2/3} \bar{\delta}^{1/3}}{-(n_\rho + n_T)} R_{iso}^{n_T} \tag{8.19}$$

We can compare τ_{analy} to that inferred from the numerical simulations using $n_\rho = \beta$ from Table 8.1 and n_T from Table 8.3. Table 8.6 summarises the results. Within error, the values for τ from the scaling law fit, τ_{num}, the τ-profile fit, τ_o, and the semi-analytic expectation assuming hydrostatic equilibrium, τ_{analy}, agree with the exception of the values for the 7 Mpc top-hat model. The trend for τ to decrease with increased smoothing is not as obvoius. In Sections 8.3 and 8.4, it is seen that n_ρ is invariant and n_T is anly weakly variant among the runs. It is the increase in R_{iso} with increased smoothing that leads to the decrease in τ_{analy} (recalling that n_T is negative). The slight increase in n_T with smoothing tends to offset this trend. Consequently, within the scatter among clusters, the value for τ does not vary appreciably and could be considered a constant among cosmologies.

Chapter 9

BARYON FRACTION

Since the early hydrodynamic simulations of galaxy clusters (Evrard 1990), the global distribution of the baryons has been of interest. In particular, the local enhancement or deficit of the baryons compared with the dark matter has been of interest since it has profound implications for inferences of the universal baryon fraction as parametrised by Ω_b. Recall that Ω is the ratio of the mass-energy density of the universe to the critical density required to close the universe. Ω_b is the contribution of the baryonic component to Ω, while Ω_m is the contribution of all mass. The fraction, Ω_b, is constrained by primordial nucleosynthesis calculations which are sensitive to the cosmological model. Currently, estimates vary between sources with lows of $\Omega_b = 0.013 \pm 0.003 h^{-2}$ (White and Fabian 1995) to $\Omega_b = (0.020 \pm 0.002) h^{-2}$ (Bludman 1998). Since galaxy clusters are the largest objects in the universe for which one can observe the baryon content as well as derive the total mass, they provide the most unbiased samples from which to calculate the baryon fraction of the universe. These values for Ω_b and the baryon fraction, f_b, may be combined to derive Ω_m, the total contribution of matter to Ω. Observations of this sort find baryon fractions of 10 to 22% (White and Fabian 1995; White, Jones, and Forman 1997). This implies $\Omega_m < 0.9$ for $\langle f_b \rangle = 0.1$ and $h = 0.5$ while suggesting it is probably closer to 0.3 if we

take $\langle f_b \rangle = 0.15$ and $h = 0.65$. However, this assumes the dark matter and hot gas are distributed in constant proportions in the cluster. Consequently, an understanding of the concentration factor of the gas in clusters is required since it represents the degree of biasing. As well, this biasing can have consequences in regards to constraining the deceleration parameter, q_0, (Rines, Forman, Pen, and Jones 1998).

Numerical simulations generally agree that the gas is anti-biased in clusters of galaxies (Evrard 1990; Thomas and Couchman 1992; Cen and Ostriker 1993; Kang, Cen, Ostriker, and Ryu 1994; Metzler and Evrard 1994; Pearce, Thomas, and Couchman 1994; Navarro, Frenk, and White 1995; Anninos and Norman 1996; Lubin, Cen, Bahcall, and Ostriker 1996; Pildis, Evrard, and Bergman 1996). Some have produced results to the contrary (Owen and Villumsen 1997). Pearce, Thomas, and Couchman (1994) explains the phenomenon as a result of the merging process in which gas is shocked, permanently removing the energy from the dark matter component. Two-body heating, a numerical artifact, can exchange energy between the dark matter and the gas (Steinmetz and White 1997) (see Sec. 4.3). This may be a factor in the outer halo of these simulations. The mean interparticle separation in the outer radii of the halos is greater than the gravitational softening length. Consequently, Eq. 4.3 applies from which it is apparent that the heating time in the outer radii is on the order of the Hubble time.

In this chapter, the dependency of the degree of biasing on the amount of hierarchical clustering and, hence, merging, will be examined. As with the other tests in this thesis, this will test the robustness of the degree of biasing to extremes in cosmology.

9.1 The Concentration Parameter, Υ

If we define $\Upsilon(\bar{\delta})$ as the gas fraction in a spherical region of overdensity $\bar{\delta}$ normalised

to the cosmic value, then we have, (White et al. 1993)

$$\Upsilon(\bar{\delta}) \equiv \frac{M_{gas}(R_{\bar{\delta}})}{M_{gas}(R_{\bar{\delta}}) + M_{dark}(R_{\bar{\delta}})} \frac{\Omega_m}{\Omega_b} \tag{9.1}$$

where Ω_m and Ω_b are the contributions to Ω from, respectively, all the matter (dark plus gas) and from just the baryons.

Similarly, we may define $\Upsilon(r)$ as the normalised gas fraction in a shell of radius r. Here, r is replaced by the dimensionless parameter $r/R_{\bar{\delta}}$ since radial profiles of both the dark matter and gas densities are characterised by the radius, $R_{\bar{\delta}}$ when $\bar{\delta}$ approximates the virial overdensity (see Sections 8.3 and 7.2).

It should be noted that the terms 'baryons' and 'gas' are interchangeable for this analysis.

9.2 Variation of Υ on Cluster Mass

The values of $\Upsilon(\bar{\delta})$, when plotted *vs.* $M_{\bar{\delta}}$ (Fig. 9.1, left), firstly support the values found in Evrard (1997) and Eke, Navarro, and Frenk (1998) of $\Upsilon(\bar{\delta} = 200) = 0.85$ to 0.90 if the sample is restricted to the largest clusters. For the lower mass clusters, the clusters in the unsmoothed model have values ranging down to 0.75. There is no trend with mass for this model other than an increase in dispersion. The mean for the hierarchically formed clusters is 0.84 with the standard deviation varying from ± 0.04 for clusters less massive than $10^{14}\,M_\odot$ to ± 0.01 for clusters more massive than $10^{15}\,M_\odot$. For the smoothed models, the trend is for Υ to decrease with mass with little change in the dispersion. For clusters more massive than $10^{15}\,M_\odot$, $\Upsilon = 0.87$ to 0.97. Below $10^{14}\,M_\odot$, this drops down to 0.82 to 0.92.

9.3 Baryon Concentration Profiles

Since it is found that both the dark matter and gas density profiles scale with the virial radius (Sections 7.2 and 8.3), it stands to reason that the radial profile of

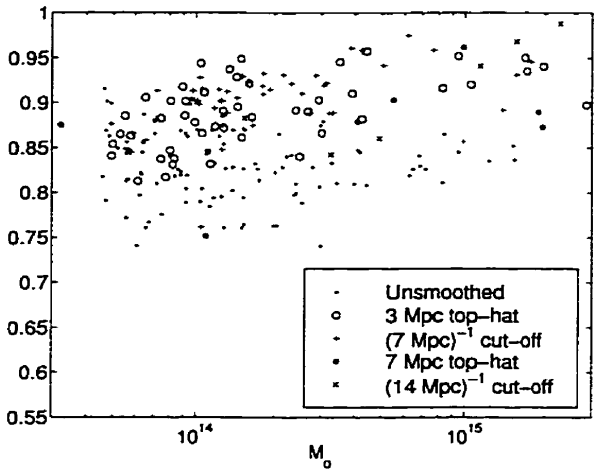


Figure 9.1: The normalised baryon fraction as a function of mass. The baryon fraction is measured by $\Upsilon(\bar{\delta} = 200)$ as a function of M_{200}.

the baryon concentration should also scale with radius. Numerical simulations of galaxy clusters using a 1-D Lagrangian code indicate that this is not true. Knight and Ponman (1997) reports that f_b varies by a factor of about two at a given scaled radius between clusters that differ in mass by two orders of magnitude, with the less massive cluster having the smaller baryon fraction. This trend is consistent with the variation in Υ with the mass of the cluster seen for the clusters formed from smoothed initial conditions. However, since this factor is comparable to the cluster to cluster variation, it is justified once again to look at the mean radial profile of all the clusters in a model.

The baryon fraction profiles vary considerably for those clusters formed in the simulation with hierarchical clustering from those formed otherwise (Fig. 9.2, left) particularly around the virial radius. The baryon fraction is enhanced in a much deeper region ($0.30 < r/R_{200} < 30$) in the hierarchical case than in the case with smoothed initial conditions ($1 < r/R_{200} < 4$) and the enhancement is almost 4 times greater. In both cases, the peak in the baryon enrichment occurs just beyond $r = R_{200}$. Towards the more central regions, the enrichment in the hierarchical clusters drops

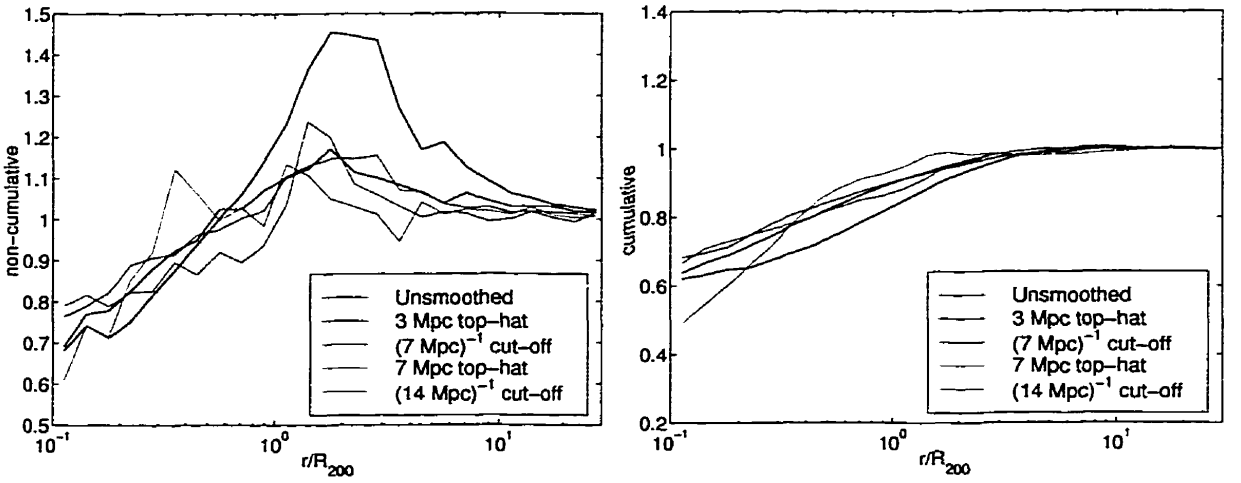


Figure 9.2: The normalised mean baryon fraction profiles, as measured by $\Upsilon(r/R_{200})$, for the models. On the left are $\langle\Upsilon\rangle$ at the specified radii. On the right are the mean cumulative enrichment profiles.

off slightly more steeply.

The discrepancies are not so large for the cumulative profile (Fig. 9.2, right). The clusters in the hierarchical model are generally more depleted in baryons than those in the smoothed models, in agreement with the results in Sec. 9.2. The mean profile for the unsmoothed model compares well those of the simulated clusters discussed in Eke, Navarro, and Frenk (1998). Those are simulated for a low-density universe.

It must be noted that there are large deviations in these profiles among the many clusters in the hierarchical scenario. These mean profiles represent mainly trends. For the clusters formed non-hierarchically, however, the baryon profiles are fairly consistent among clusters.

For the outer radii of the mean Υ profile for the unsmoothed model, it is found that $\Upsilon(r) = 0.48 \pm 0.03(1 + 0.7 \pm 0.1r/R_{200})$, non-cumulative. For all data sets, the profiles are together fit by a shallower form, $\Upsilon(r) = 0.58 \pm 0.02(1 + 0.24 \pm 0.1r/R_{200})$

The determination of baryon fraction profiles of observed clusters via the deprojection of the gas density (White, Jones, and Forman 1997) imaged in the X-ray regime

supports the results here (Markevitch and Vikhlinin 1997b). White and Fabian (1995) finds from a sample of 19 clusters that the cumulative measurement of f_b increases with radius following roughly the locus $f_b = 0.06(1 + R)$ with R in Mpc. Similarly, White, Jones, and Forman (1997) fits $f_b = (0.12 \pm 0.04)R^{0.7\pm0.3}$ for a sample of 28 clusters without cooling flows. The sample of clusters with cooling flows has a higher central f_b and a shallower profile, as would be expected. The fractional slopes of both these fits are steeper than that of the mean Υ profile of all the models. However, the fractional slope of Υ for the unsmoothed model is comparable with the results of White and Fabian (1995).

9.4 Discussion

These results indicate that the observed baryon fractions in galaxy clusters are underestimating the universal baryon fraction by a factor of 5 to 20% and this factor is dependent on the cosmology. The anti-biasing of the gas is greatest in the hierarchical run, consistent with the picture of Pearce, Thomas, and Couchman (1994). Since cooling flows are not created in these simulations, this factor is not well established here. However, cooling flows will only alter the baryon fractions in the inner radii since the total mass cooled is only on the order of $10^{12}\,M_\odot$ or less. Ultimately, this exacerbates the baryon overdensity problem in galaxy clusters. The variation in the enrichment of baryons with distance from the cluster centres also makes measurements of f_b problematic. It is crucial that the cluster be observed to large radii.

Chapter 10

CONCLUSIONS

This thesis finds the overall morphology of the hot, gaseous component of clusters of galaxies robust to their formation method and relatively independent of the hierarchical clustering nature of structure formation. The dark matter profile, the mass-temperature relationship, and the baryon fraction of the clusters do not vary substantially among clusters formed in hierarchical structure formation scenarios and those formed non-hierarchically. However, these features do vary in ways which may be significant to both theory and observations. In short, both the theorists and the observers may rest easy. More detailed conclusions follow, but first the results of the parallelisation of the code used for these simulations as well as the results of the tests of the hydrodynamical method used.

10.1 Parallelisation of HYDRA

A method for parallelising FORTRAN 77 code using Pthreads has been described. It requires minimal modification to the algorithms. The method involves using C functions as intermediaries to the FORTRAN subroutines to be executed in parallel. The N-body code with hydrodynamics, HYDRA, was parallelised using the technique described. The modifications provide $> 60\%$ effective parallelisation on a two processor

UltraSPARC which does not compare well with more machine-dependent implementations but is an acceptable return considering the ease of implementation. Moreover, the portability of the technique has been demonstrated by the use of the code on an i686 platform operating the Linux operating system; the parallelised code compiled and ran 'out of the box'.

10.2 Tests of SPH

It has been shown that the smoothed particle hydrodynamics (SPH) inference of the density of a particle overestimates the density significantly ($> 20\%$) if the true density changes across a distance equal to twice the smoothing length of the particle by more than 3 times. This allows the derivation of an approximate radius interior to which the densities estimated must be considered erroneous. For a given number density profile with the dependency $n(r) = n_o r^{-\alpha}$, the critical radius is approximated by

$$r_c \simeq \frac{2\alpha}{(\delta_c N)^{1/3}}, \tag{10.1}$$

where δ_c is the central overdensity and N is the number of particles in the simulation. Furthermore, this radius corresponds to a minimum number of particles,

$$N_{min} \simeq 4(2\alpha)^3. \tag{10.2}$$

This inability of SPH to model arbitrarily steep density gradients leads to an overcooling phenomenon in cosmological simulations. It is found that this phenomenon is endemic to a variety of implementations of SPH.

Tests examining the drag of a cold particle through a hot gas reveal that this is not sensitive to the form of the symmetrisation of the equations of motion in the implementation of SPH, nor is it sensitive to the form of the artificial viscosity. However, the inclusion of the artificial viscosity reduces the drag due to a reduction in

the amount of interpenetration of particles that the viscosity provides. The inclusion of a shear-correction term enhances beneficially this effect.

10.3 Hydrostatic State of Clusters

The clusters of galaxies simulated for this work are shown to be in hydrostatic equilibrium to beyond their virial radius as indicated by R_{200}. At this radius, the ratio of the actual gas pressure to the gas pressure expected, given the mass distribution, is found to be within 20% of unity for the majority of the clusters.

10.4 Universal Density Profile

The density profile form of Navarro, Frenk, and White is found to fit the mean dark matter density profiles of the clusters of all models studied, indicating that *the universal profile is not a by-product of hierarchical clustering.* This profile form, however, fits only marginally better than the Hernquist profile. Indeed, the discontinuous form suggested here generally fits as well, if not better, than the NFW form. However, it has more free parameters. The discontinuous form indicates that the inner regimes of the density profile are approximately dependent on radius as $\rho \propto r^{-1.8}$, and independent of the degree of smoothing of the initial conditions. The outer regimes, in contrast, do seem to be dependent on smoothing. The profile becomes more shallow as the degree of smoothing is increased. The dependency varies from $\rho \propto r^{-2.7}$ in the case of the hierarchical, unsmoothed case to $\rho \propto r^{-2.3}$ for the 7 Mpc top-hat smoothed model and the $(14Mpc)^{-1}$ frequency cut-off model.

10.5 Mass-Temperature Scaling Law

The mass-temperature scaling law (Eq. 8.17) is dependent on the degree of hierarchical cluster formation, albeit at a 1σ level. The temperature parameter τ_{num} varies from 7 ± 1 in the unsmoothed case to 5 ± 1 in case of the 7 Mpc top-hat run with the intermediate smoothing models falling in between. The trend is consistent to the

expectation of hydrostatic equilibrium. The expected value, τ_{analy}, decreases from 7.7 ± 0.2 to 6.4 ± 0.7 over the span of smoothing. However, there is a systematic discrepancy between the expectation, τ_{analy}, and the numeric, τ_{num}, which increases with the amount of smoothing. The results of the fits to the profiles indicates that it is the variation in the isothermal radius that explains the variation with smoothing. The isothermal radius increases as the degree of hierarchical clustering is reduced. This occurs in step with the gas density profiles which become shallower in the inner radii.

10.6 Baryon Fraction Bias

It is found that the baryons are anti-biased in clusters of galaxies and this biasing is dependent on the presence of smoothing in the initial conditions. The bias parameter $\Upsilon \simeq 0.85$ for the hierarchically formed clusters. For the clusters formed from smoothed initial conditions, $\Upsilon \simeq 0.92$, indicating less of an anti-bias. For the cluster formed hierarchically, the baryons 'pile up' in a deep region spanning $0.30 < r/R_{200} < 30$. This implies that measurements of the baryon fraction of clusters are sensitive to the radius outward to which the baryon content is integrated.

10.7 Future Work

Higher resolution studies are always beneficial, if not to fine-tune the results, then to give further support. Performing the set of simulations with another code would also be educational.

As suggested in Chapter 9, two-body interactions can transfer energy between the dark matter and the gas in the outer halos of the clusters. This numerical heating may be affecting the baryon fractions in these regimes. A solution to this problem is not obvious, as increasing the gravitational softening length, which suppresses two-body interactions, dramatically decreases the efficiency and effective resolution in the higher density regions. One could wish for a softening length that varies in space in

a manner similar to the SPH smoothing length, but this does not conserve energy.

More work is required to determine the parameter space in which SPH produces the overcooling phenomenon. Similar work needs to be done for the drag experienced by a cold clump moving through a hot medium, as well as determining precisely *why* the clump experiences such a drag. In both cases, domains were selected which were known to produce the undesirable effects. Little was done to delineate this domain.

The question of energy transfer between the dark matter and the gas is left unexplored. The answer to whether the presence of substructure enhances this transfer may explain the variation of the gas profiles, beyond the gas's response to the different dark matter distributions which formed hierarchically.

10.8 What it all means

The NFW profile has been established to be a robust description of the dark matter density profile, independent of hierarchical clustering and, by extension, independent of the assumed cosmology. This strongly implies that dark matter in galaxy clusters is distributed in this manner. If detailed mass maps of clusters using gravitational lensing observations indicate that the mass is not distributed in this fashion, it will indicate that either numerical simulations are missing some significant physics or the Standard Model is flawed in a fundamental manner. It will be difficult to attribute the discrepancy to the assumed initial density perturbation spectrum within the Standard Model. Unfortunately, this also implies that the observed mass density profiles of clusters will tell us little about this initial perturbation spectrum.

A similar conclusion can be made for the mass-temperature relation for clusters of galaxies. Fortunately, there is already support from the observations of clusters for this relation. There is a lack, however, of a calibration between the observed mass-temperature relation and that inferred from numerical simulations. The simulations

may be missing some physics, such as star formation feedback, that can be expected to modify the temperature of the cluster gas. The form, itself, is robust, and independent of the initial perturbation spectrum.

Observations of the baryon fraction in clusters will be hampered by the biased distribution of gas in clusters reported in this thesis. Measurements must be made to more than twice the virial radius, sampling regions for which accurate densities of both mass and gas are difficult. This uneven distribution is not as significant if there is any degree of smoothing, implying that any attempt to formulate a correction factor will be dependent on the cosmology at the 10% level. Fortunately, this is at the present level of measurement error.

Bibliography

Allen, S. & Fabian, A., 1997. *MNRAS* **286**, 583.

Anninos, P. & Norman, M., 1996. *ApJ* **459**, 12.

Balogh, M. L., Babul, A., & Patton, D. R., 1998, astro-ph/9809159.

Balsara, D. S., 1995. *Journal of Computational Physics* **121**, 357.

Bertschinger, E., 1994. *Physica D* **77**, 354.

Bludman, S., 1998. *ApJ* **508**, 535.

Bryan, G. & Norman, M., 1998. *ApJ* **495**, 80.

Cavaliere, A. & Fusco-Femiano, R., 1976. *AA* **49**, 137.

Cen, R. & Ostriker, J., 1993. *ApJ* **417**, 404.

Couchman, H., Thomas, P., & Pearce, F., 1995. *ApJ* **452**, 797.

Crone, M., Evrard, A., & Richstone, D., 1994. *ApJ* **434**, 402.

Efstathiou, G. & Eastwood, J., 1981. *MNRAS* **194**, 503.

Eke, V., Navarro, J., & Frenk, C., 1998. *ApJ* **503**, 569.

Evans, N. & Collett, J., 1997. *ApJL* **480**, 103.

Evrard, A., 1988. *MNRAS* **235**, 911.

Evrard, A., 1990. *ApJ* **363**, 349.

Evrard, A., 1997, astro-ph/9701148.

Evrard, A., Metzler, C., & Navarro, J., 1996. *ApJ* **469**, 494.

Fabricant & Gorenstein, 1983. *ApJ* **267**, 535.

Frenk, C. & *et al*, 1998, In preperation.

Frenk, C. S., Evrard, A. E., White, S. D. M., & Summers, F. J., 1996. *ApJ* **472**, 460.

Fukushige, T. & Makino, J., 1997. *ApJL* **477**, 9.

Gingold, R. A. & Monaghan, J. J., 1977. *MNRAS* **181**, 375.

Henriksen, R. & Widrow, L., 1998, astro-ph/9805277.

Hernquist, L., 1990. *ApJ* **356**, 359.

Hoffman, Y. & Shaham, J., 1985. *ApJ* **297**, 16.

Huss, A., Jain, B., & Steinmetz, M., 1998, astro-ph/9803117.

Jones, C. & Forman, W., 1984. *ApJ* **276**, 38.

Kang, H., Cen, R., Ostriker, J., & Ryu, D., 1994. *ApJ* **428**, 1.

Knight, P. & Ponman, T., 1997. *MNRAS* **289**, 955.

Lombardi, J., Sills, A., & Shapiro, S., 1998, astro-ph/9807290.

Lubin, L., Cen, R., Bahcall, N., & Ostriker, J., 1996. *ApJ* **460**, 10.

Lucy, L., 1977. *AJ* **82**, 1013.

Markevitch, M. & Vikhlinin, A., 1997a. *ApJ* **474**, 84.

Markevitch, M. & Vikhlinin, A., 1997b. *ApJ* **491**, 467.

Metzler, C. & Evrard, A., 1994. *ApJ* **437**, 564.

Monaghan, J. J., 1992. *ARAA*, pp. 543. Palo Alto, California: Annual Reviews Inc.

Moore, B., Ghigna, S., Governato, F., Lake, G., Quinn, T., & Stadel, J., 1997, astro-phy/9711259.

Moore, B., Governato, F., Quinn, T., Stadel, J., & Lake, G., 1998. *ApJL* **499**, 5.

Moore, B., Katz, N., Lake, G., Dressler, A., & Oemler, A., 1996. *Nature* **379**, 613.

Navarro, J., Frenk, C., & White, S., 1995. *MNRAS* **275**, 720.

Navarro, J., Frenk, C., & White, S., 1996. *ApJ* **462**, 563.

Navarro, J., Frenk, C., & White, S., 1997. *ApJ* **490**, 493.

Owen, J. & Villumsen, J., 1997. *ApJ* **481**, 1.

Padmanabhan, T., 1996. *MNRAS* **278**, 29.

Padmanabhan, T., Cen, R., Ostriker, J., & Summers, F., 1996. *ApJ* **466**, 604.

Pearce, F. & Couchman, H., 1997. *New Astro.* **vol. 2 num. 5**, 411.

Pearce, F., Thomas, P., & Couchman, H., 1994. *MNRAS* **268**, 953.

Pildis, R., Evrard, A., & Bergman, J., 1996. *AJ* **112**, 378.

Rines, K., Forman, W., Pen, U., & Jones, C., 1998, astro-ph/9809336.

Schindler, S., 1996. *AA* **305**, 756.

Steinmetz, M., 1996. *MNRAS* **278**, 1005.

Steinmetz, M. & Müller, E., 1993. *AA* **268**, 391.

Steinmetz, M. & White, S. D., 1997. *MNRAS* **288**, 545.

Suginohara, T. & Ostriker, J. P., 1998. *ApJ* **507**, 16.

Syer, D. & White, S., 1998. *MNRAS* **293**, 337.

T.G. Brainerd, D.M. Goldberg, J. V., 1998. *ApJ* **502**, 505.

Thacker, R. J., Couchman, H. M. P., & Pearce, F. R., 1998. *High Performance Computing Systems and Applications*, pp. 213. Kluwer Academic.

Thacker, R. J., Tittley, E. R., Pearce, F. R., Couchman, H. M. P., & Thomas, P. A., 1998, astro-ph/9809221.

Thomas, P., Colberg, J., Couchman, H., Efstathiou, G., Frenk, C., Jenkins, A., Nelson, A., Hutchings, R., Peacock, J., Pearce, F., & White, S., 1998. *MNRAS* **296**, 1061.

Thomas, P. A. & Couchman, H. M. P., 1992. *MNRAS* **257**, 11.

Tsai, J. C., Katz, N., & Bertschinger, E., 1994. *ApJ* **423**, 553.

White, D. & Fabian, A., 1995. *MNRAS* **273**, 72.

White, D., Jones, C., & Forman, W., 1997. *MNRAS* **292**, 419.

White, S., Navarro, J., Evrard, A., & Frenk, C., 1993. *Nature* **366**, 429.

Zel'dovich, Y. B., 1970. *AA* **5**, 84.