Research in Astron. Astrophys. **2009** Vol. **9** No. **8**, 861 – 873
http://www.raa-journal.org *http://www.iop.org/journals/raa*

***R**esearch in*
***A**stronomy and*
***A**strophysics*

A modified TreePM code

Nishikanta Khandai and J. S. Bagla

Harish-Chandra Research Institute, Chhatnag Road, Jhusi, Allahabad 211019, India; *nishi@hri.res.in, jasjeet@hri.res.in*

Received 2009 January 31; accepted 2009 April 23

Abstract We discuss the performance characteristics of using the modification of the tree code suggested by Barnes in the context of the TreePM code. The optimization involves identifying groups of particles and using only one tree walk to compute the force for all the particles in the group. This modification has been in use in our implementation of the TreePM code for some time, and has also been used by others in codes that make use of tree structures. We present the first detailed study of the performance characteristics of this optimization. We show that the modification, if tuned properly, can speed up the TreePM code by a significant amount. We also combine this modification with the use of individual time steps and indicate how to combine these two schemes in an optimal fashion. We find that the combination is at least a factor of two faster than the modified TreePM without individual time steps. Overall performance is often faster by a larger factor because the scheme for the groups optimizes the use of cache for large simulations.

Key words: cosmology: large-scale structure of universe — gravitation — methods: numerical

1 INTRODUCTION

Large scale structures traced by galaxies are believed to have been formed by amplification of small perturbations (Peebles 1980; Peacock 1999; Padmanabhan 2002; Bernardeau et al. 2002). Galaxies are highly over-dense systems; matter density ρ in galaxies is thousands of times larger than the average density $\bar{\rho}$ in the universe. The typical density contrast ($\delta \equiv \rho/\bar{\rho} - 1$) in matter at these scales in the early universe was much smaller than unity. Thus, the problem of galaxy formation and the large scale distribution of galaxies requires an understanding of the evolution of density perturbations from small initial values to the large values we encounter today.

Initial density perturbations were present at all scales that have been observed (Spergel et al. 2007; Percival et al. 2007). The equations that describe the evolution of density perturbations in an expanding universe have been known for several decades (Peebles 1974) and these are easy to solve when the amplitude of perturbations is small. Once density contrast at relevant scales becomes comparable to unity, perturbations become non-linear and coupling with perturbations at other scales cannot be ignored. The equation for evolution of density perturbations cannot be solved for generic initial conditions in this regime. N-body simulations (e.g., see Efstathiou et al. 1985; Bertschinger 1998; Bagla & Padmanabhan 1997; Bagla 2005) are often used to study the evolution in this regime. Alternative approaches can be used if one requires only a limited amount of information and in such a case either using quasi-linear approximation schemes (Bernardeau et al. 2002; Zel'Dovich 1970; Gurbatov, Saichev & Shandarin 1989; Matarrese et al. 1992; Brainerd, Scherrer & Villumsen 1993; Bagla & Padmanabhan 1994; Sahni & Coles 1995; Hui & Bertschinger 1996) or scaling relations (Davis & Peebles 1977; Hamilton et al. 1991; Jain, Mo & White 1995; Kanekar 2000; Ma 1998; Nityananda & Padmanabhan

1994; Padmanabhan et al. 1996; Peacock & Dodds 1994; Padmanabhan 1996; Peacock & Dodds 1996; Smith et al. 2003) suffice. However, even the approximation schemes and scaling relations must be compared with simulations before these can be used with confidence.

The last three decades have seen a rapid development of techniques and computing power for cosmological simulations and the results of these simulations have provided valuable insight into the study of structure formation. State of the art simulations used less than 10^5 particles two decades ago (Efstathiou et al. 1988) and if the improvement had been due only to computing power then the largest simulation possible today should have been around 10^9 particles, whereas the largest simulations done to date have used more than 10^{10} particles (Springel et al. 2005). Evidently, the development of new methods and optimizations has also played a significant role in the evolution of simulation studies (Efstathiou et al. 1985; Barnes & Hut 1986; Greengard & Rokhlin 1987; Bouchet & Hernquist 1988; Jernigan & Porter 1989; Hernquist 1990; Makino 1990, 1991; Hernquist, Bouchet & Suto 1991; Couchman 1991; Ebisuzaki et al. 1993; Theuns 1994; Brieu, Summers & Ostriker 1995; Suisalu & Saar 1995; Xu 1995; Dubinski 1996; Kravtsov, Klypin & Khokhlov 1997; Macfarland et al. 1998; Bode, Ostriker & Xu 2000; Brieu & Evrard 2000; Dehnen 2000; Knebe, Green & Binney 2001; Springel, Yoshida & White 2001; Kawai & Makino 2001; Makino 2002; Dehnen 2002; Bagla 2002; Bagla & Ray 2003; Makino et al. 2003; Bode & Ostriker 2003; Ray & Bagla 2004; Dubinski et al. 2004; Makino 2004; Springel 2005; Merz, Pen & Trac 2005; Yoshikawa & Fukushige 2005; Wadsley, Stadel & Quinn 2004; Thacker & Couchman 2006). Along the way, code developers have also successfully met the challenge posed by the emergence of distributed parallel programming.

In this paper, we discuss the performance characteristics of an optimization for tree codes suggested by Barnes (1990). We do this in the context of the TreePM method (Bagla 2002; Bagla & Ray 2003) where the tree method is used for computing the short-range force. The TreePM method brings an additional scale into the problem, i.e., the scale up to which the short-range force is computed and this leads to non-trivial variations of error in force.

The paper is organized as follows: we introduce the TreePM method in Section 2, and discuss the optimization scheme in Section 3. Performance of the optimization scheme is discussed in Section 4, and we discuss combining this with individual time steps for particles in Section 5. We end with a discussion in Section 6.

2 THE TREEPM ALGORITHM

The TreePM algorithm (Bagla 2002; Bagla & Ray 2003) is a hybrid N-body method which combines the BH-Tree method (Barnes & Hut 1986) with the PM method (Bagla & Padmanabhan 1997; Merz, Pen & Trac 2005; Klypin & Shandarin 1983; Miller 1983; Bouchet & Kandrup 1985; Bouchet, Adam & Pellat 1985; Hockney & Eastwood 1988). The TreePM method explicitly breaks the potential into a short-range and a long-range component at a scale r_s. The PM method is used to calculate the long-range force and the short-range force is computed using the BH-Tree method. Use of the BH Tree for short-range force calculations enhances the force resolution as compared to the PM method.

The gravitational force is divided into a long-range and a short-range part using partitioning of unity in the Poisson equation.

$$\begin{aligned}
\phi_k &= -\frac{4\pi G\rho_k}{k^2} \\
&= -\frac{4\pi G\rho_k}{k^2}\exp\left(-k^2r_s^2\right) - \frac{4\pi G\rho_k}{k^2}\left[1-\exp\left(-k^2r_s^2\right)\right] \\
&= \phi_k^{\rm lr} + \phi_k^{\rm sr} \\
\phi_k^{\rm lr} &= -\frac{4\pi G\rho_k}{k^2}\exp\left(-k^2r_s^2\right), \qquad (1) \\
\phi_k^{\rm sr} &= -\frac{4\pi G\rho_k}{k^2}\left[1-\exp\left(-k^2r_s^2\right)\right]. \qquad (2)
\end{aligned}$$

Here $\phi_k^{\rm sr}$ and $\phi_k^{\rm lr}$ are the short-range and long-range potentials in Fourier space. ρ is the density, G is the gravitational coupling constant and r_s is the scale at which the splitting of the potential is done. The long-range force is solved in Fourier space with the PM method and the short-range force is solved in real space with the Tree method. The short-range force in real space is

$$\boldsymbol{f}^{\rm sr}(\boldsymbol{r}) = -\frac{GM\boldsymbol{r}}{r^3}\left[{\rm erfc}\left(\frac{r}{2r_s}\right) + \frac{r}{r_s\sqrt{\pi}}\exp\left(-\frac{r^2}{4r_s^2}\right)\right], \tag{3}$$

where 'erfc' is the complementary error function.

The short-range force is below 1% of the total force at $r \geq 5\,r_s$. The short-range force is therefore computed within a sphere of radius $r_{\rm cut} \simeq 5\,r_s$. The short range force is computed using the BH-Tree method. The tree structure is built out of cells and particles. Cells may contain smaller cells (subcells) within them. Subcells can have even smaller cells within them, or they can contain a particle. In three dimensions, each cubic cell is divided into eight cubic subcells. Cells, as structures, have attributes like total mass, location of the center of mass and pointers to subcells. Particles, on the other hand, have the usual attributes: position, velocity and mass.

Force on a particle is computed by adding the contributions of other particles or of cells. A cell that is sufficiently far away can be considered as a single entity and we can add the force due to the total mass contained in the cell from its center of mass. If the cell is not sufficiently far away then we must consider its constituents, subcells and particles. Whether a cell can be accepted as a single entity for force calculation is decided by the cell acceptance criterion (CAC). We compute the ratio of the size of the cell d and the distance r from the particle in question to its center of mass and compare it with a threshold value

$$\theta = \frac{d}{r} \leq \theta_c. \tag{4}$$

The error in force increases with θ_c. A poor choice of θ_c can lead to significant errors (Salmon & Warren 1994). Many different approaches have been tried for the CAC in order to minimize error as well as CPU time usage (Salmon & Warren 1994; Springel, Yoshida & White 2001). The tree code gains over direct summation as the number of contributions to the force becomes much smaller than the number of particles.

The TreePM method is therefore characterized by three parameters, r_s, $r_{\rm cut}$ and θ_c. For a discussion of the optimum choice of these parameters the reader is referred to Bagla & Ray (2003).

3 THE SCHEME OF GROUPS

We first describe an optimization scheme due to Barnes (1990), given in the paper with a curious title *A modified tree code. Don't laugh, it runs.* This scheme is easily portable to any N-body algorithm that uses tree data structures to compute forces. The origin of the optimization is in the realization that the tree walk used for computing forces is computationally the most expensive component of a tree code. The idea is to have a common interaction list for a *group* of particles that is sufficiently small. Given that we are working with a tree code, it is natural to identify a cell in the tree structure as a group. One can then add the contribution of particles within the group using direct pair summation. The cell acceptance criterion (CAC) for the tree walk needs to be modified in order to take the finite size of the group into account. In our implementation of the TreePM method, we modified the standard CAC in the following manner:

$$d \leq (r - r_m)\,\theta_c, \tag{5}$$

where r_m is the distance between the center of mass of the group and the group member that is farthest from the center of mass. This is calculated once before the force calculation and does not add much in terms of overhead.

The modified CAC can be thought of as the standard CAC with a distance dependent θ_c, with the value of θ_c decreasing at small r. As we require a larger number of operations for smaller θ_c, each tree walk with the modified CAC is expected to require more CPU time than a tree walk with the standard

CAC. However, as we do a tree walk for a group of particles in one go, CPU time is saved as the time taken for a tree walk per particle comes down.

There is an overhead as there is a pair-wise force calculation within the group. The cost of this overhead increases as the square of the number of particles in the group. In order to keep the overhead small, one would like the group to be sufficiently small compared to the size of the N-body simulation and hence a maximum size $c_{\rm max}$ and an upper bound on the number of particles in the group $n_{\rm pmax}$ are used. An upper limit on the size of the group is pertinent because of the indirect effect through the change in the CAC. The effect of the additional parameter $c_{\rm max}$ with the modified CAC will be seen when we discuss errors in Section 4. Our implementation of the modified method by using a different definition of groups, with the additional parameter $c_{\rm max}$ and the modified CAC (Eq. (5)) ensures that the short-range force is extremely accurate. This is different from previous implementations (Barnes 1990; Makino 1991; Yoshikawa & Fukushige 2005; Wadsley, Stadel & Quinn 2004) where the group scheme was parameterized by just one parameter $n_{\rm pmax}$ and the standard CAC (Eq. (4)) used for tree traversal. We note in passing that the modified CAC is crucial in order to limit errors. Indeed, we find that working with the standard CAC leads to errors in short-range force that are orders of magnitude larger.

3.1 Estimating Speedup

We model the modified Tree/TreePM method with the aim of estimating the speedup that can be achieved. If N is the total number of particles, $n_{\rm p}$ the typical number of particles in a group and $n_{\rm g}$ the number of groups then clearly we expect $n_{\rm g} \times n_{\rm p} = N$. The total time required for force calculation is a sum of the time taken up by the tree walk and the time taken up by pairwise calculation within the group. Actual calculation of the force, once the interaction list has been prepared, takes very little time and can be ignored in this estimate, as can the time taken to construct the tree structure. The time taken is:

$$T_g = \alpha n_{\rm g} \ln N + \beta n_{\rm g} n_{\rm p}^2 = \alpha \frac{N}{n_{\rm p}} \ln N + \beta N n_{\rm p}. \tag{6}$$

Here we have assumed that the time taken per tree walk scales as $\mathcal{O}(\ln N)$ even with the modified CAC [1]. The time taken is smallest when

$$n_{\rm p} = \left(\frac{\alpha \ln N}{\beta} \right)^{1/2}; \qquad T_{g_{\rm min}} = 2\beta N n_{\rm p} = 2\alpha \frac{N}{n_{\rm p}} \ln N. \tag{7}$$

Thus, the optimum number of particles in the group scales weakly with the total number of particles. In the optimal situation, we expect the tree walk and the pairwise components to take the same amount of CPU time.

For comparison, the time taken for force calculation in the standard TreePM is:

$$T = \alpha N \ln N, \tag{8}$$

and we make the simplifying assumption that α is the same for the two cases. The expected speedup is then given by:

$$\frac{T}{T_{g_{\rm min}}} = \frac{1}{2} \left(\frac{\alpha \ln N}{\beta} \right)^{1/2}. \tag{9}$$

The speedup for the optimum configuration scales in the same manner as the typical number of particles per group.

A more detailed analysis of this type can be found in Makino (1991).

[1] This is an approximation as we expect the tree walk to depend on $c_{\rm max}$, $n_{\rm pmax}$ and θ_c as well. The finite size of groups should lead to deviations from the $\mathcal{O}(\ln N)$ variation and the deviation should scale as the ratio of the volume of the group and the volume of the simulation box. As this ratio becomes smaller for large simulation boxes, we feel that the approximation we have made is justified.

The calculation we have presented above is approximate and ignores several factors; some of these have already been highlighted above. There are other subtleties like the role played by the finite range $r_{\rm cut}$ over which the short-range force is calculated. The size of a group ($c_{\rm max}$) cannot be varied continuously, and hence $n_{\rm p}$ is also restricted to a range of values. Further, the number of operations does not translate directly into CPU time as some calculations make optimal use of the capabilities of a CPU while others do not. For example, the pairwise calculation is likely to fare better on processors with a deep pipeline for execution whereas a tree walk cannot exploit this feature. The finite bandwidth of the CPU-memory connection also has an impact on the scaling with N for large N. In the following section, we discuss the implementation of the modified TreePM method and the timing of the code with different values of parameters.

4 A MODIFIED TREEPM ALGORITHM WITH THE SCHEME OF GROUPS

Tests of the TreePM method have shown that 95%–98% of the time goes into the short-range force calculation. Keeping this in mind, the scheme of groups was introduced to optimize the short-range force calculation in terms of speed. A welcome feature is more accurate force computation. Since the optimum set of TreePM parameters has been discussed in Bagla & Ray (2003), we now look for the optimum choice of the additional parameters, $c_{\rm max}$ and $n_{\rm pmax}$, which describe the modified TreePM algorithm. The analysis that follows is divided into two parts. First we look at the optimum values of $c_{\rm max}$ and $n_{\rm pmax}$ which minimize the time for short-range force computation. Second, we study errors in total and short-range forces with this new scheme.

4.1 Optimum Parameters of the Modified TreePM Algorithm

We choose $r_s = 1$, $r_{\rm cut} = 5.2\,r_s$ and $\theta_c = 0.5$ for the discussion that follows. With this choice, the error in force for 99% of the particles is less than a few percent (Bagla & Ray 2003). We present an analysis of the performance of the modified TreePM for two different particle distributions taken from an $N-$body simulation, with $N = N_{\rm box}^3 = 200^3$.

- An unclustered distribution that corresponds to the initial conditions of an $N-$body simulation.
- A clustered distribution taken again from the same $N-$body simulation. The scale of non-linearity for the clustered distribution is eight grid lengths.

We have verified that the nature of the results does not change significantly for simulations with the number of particles ranging from 32^3 to 256^3.

In Figure 1, we show the time taken for computing the short-range force (solid line) and determine the values of $(c_{\rm max}, n_{\rm pmax})$ for which this timing is a minimum. Two leading contributions to the calculation of short-range force are shown separately:

- Intra-group particle-particle contribution (dashed line).
- Time taken for the tree-walk and the related force calculation (dot-dashed line).

Given that a group is a cell with maximum width

$$c_{\rm max} = \frac{N_{\rm box}}{2^m}, \quad (m \text{ is an integer}), \tag{10}$$

$c_{\rm max}$ can therefore only take discrete values. We choose to restrict up to $c_{\rm max} \sim 2\,r_{\rm cut}$. As for larger cells, the dominant contribution to force on a given particle arises from the intra-group particle-particle interaction and the time taken for this is a sensitive function of the amplitude of clustering.

The time taken for computing the short-range force in both, unclustered (left panel) as well as clustered (right panel), distributions is qualitatively described by our model (see Eq. (6)). The pairwise force increases linearly with $n_{\rm pmax}$, where $n_{\rm pmax}$ is the maximum number of particles in a group and scales as $n_{\rm p}$ which is the average number of particles in a group. The time taken for the tree-walk

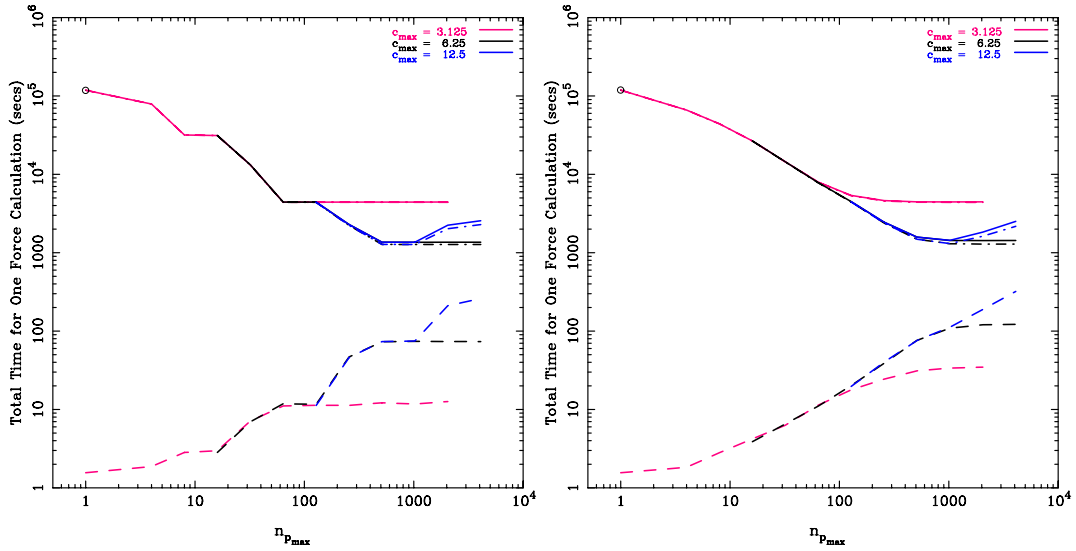

Fig. 1 Time taken for computation of the short term force in the modified TreePM method for an unclustered (*left panel*) and a clustered (*right panel*) distribution. Solid lines represent the time taken by a complete short-range force calculation. Dashed lines are the contribution to the force due to pairs within a group, the intra-group contribution. Dot-dashed lines are the contributions to force due to the tree walk. Purple, black and blue lines are for $c_{\rm max} = 3.125, 6.25$ and 12.5, respectively.

decreases as $n_{\rm pmax}^{-0.65}$, reaches a minimum and then increases with $n_{\rm pmax}$ (blue line) for the largest $c_{\rm max}$ used here. For other values of $c_{\rm max}$, we see the timing leveling off near the minimum. The scaling as $n_{\rm pmax}^{-0.65}$ is different from $1/n_{\rm p}$ which we used in the analytical model and the reason for this is likely to be in the approximations we used. We find that the scaling approaches $1/n_{\rm pmax}$ as we consider simulations with a larger number of particles. One crucial reason for the different scaling is the modified CAC we use here. This effectively leads to a smaller θ_c for cells closer to the group and the number of such cells increases with $c_{\rm max}$.

In both cases, the total time is still dominated by the treewalk. The plateaus in the plots often indicate that the number of particles in a group of maximum size $c_{\rm max}$ has been saturated. At initial times where the fluctuations are small, there is also a lower bound on the number of particles contained in a group. In the clustered distribution there is no such lower bound, but an upper bound, larger than the corresponding upper bound in the case of the unclustered distribution, exists and is dictated by the amplitude of clustering in the distribution of particles.

From Figure 1, we see that the optimum values of ($c_{\rm max}, n_{\rm pmax}$) = ($12.5$, 1024) & (6.25, ≥ 1024) given by the minima of the solid blue line and the plateau of the solid black line respectively. In the latter case, the time taken does not change for $n_{\rm pmax} \geq 1024$ and we consider this to be a useful feature that makes $c_{\rm max} = 6.25$ a better choice as fine tuning of $n_{\rm pmax}$ is not required. For the optimum ($c_{\rm max}$, $n_{\rm pmax}$), one can see that force computation takes the same time for the clustered and the unclustered distributions. Table 1 lists optimum values for ($c_{\rm max}$, $n_{\rm pmax}$) for N-body simulations with different numbers of particles. These numbers indicate that a good choice for $c_{\rm max}$ is one which is closest to $r_{\rm cut}$, i.e. $c_{\rm max} \sim r_{\rm cut}$. The parameter $n_{\rm pmax}$ can be taken to be $10^3 \leq n_{\rm pmax}$ as we find little variation beyond this point.

One can get an estimate of the overheads for the group scheme by looking at the limit of $n_{\rm pmax} \rightarrow 1$. Here we compare the performance of the TreePM with the modified code by plotting the time taken by the former as a large dot on the same panel where the time taken for the modified code is shown in the form of curves. The difference between these timings is around 0.1%.

Table 1 Optimum values of $(c_{\max}, n_{\rm pmax})$ for simulations of various sizes which have the same TreePM parameters: $r_s = 1$, $r_{\rm cut} = 5.2\, r_s$, $\theta_c = 0.5$.

$N_{\rm box} = N^{1/3}$	$c_{\max}^{\rm opt}$	$n_{\rm pmax}^{\rm opt}$
64	4.0	$\geq$1024
128	4.0	$\geq$1024
160	5.0	$\geq$1024
200	6.25	$\geq$1000

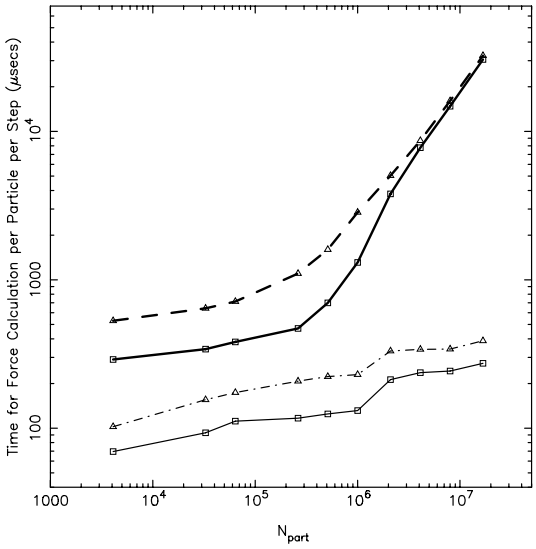

Fig. 2 Time taken for short-range force calculation per particle per step for $N = 32^3$ to $N = 256^3$ for the TreePM (thick line) and the Modified TreePM (thin line). The solid line shows the performance of the codes on a single core of an Intel 5160 (3.0 GHz) processor and the dashed line shows the performance on a single core of the AMD Barcelona (2.1 GHz) processor.

The speedup for the optimal configuration of the modified TreePM, as compared with the base TreePM code, is ~ 83. This is a huge gain and has to do with better utilization of the CPU cache. The speedup is less impressive for smaller simulations, and is larger for bigger simulations. This is shown in Figure 2 where we plot the time taken for force calculation per particle per step as a function of the total number of particles in the simulation. This is shown for the TreePM as well as the modified TreePM codes. Performance on two different types of processors is shown here to demonstrate that the optimization works equally well on both. One can see that the TreePM code becomes (CPU-memory) bandwidth limited for simulations with more than 64^3 particles and the time taken increases more rapidly than $\mathcal{O}(\ln N)$. This does not happen in case of the modified TreePM where the scaling is $\mathcal{O}(\ln N)$ throughout. It is this difference that leads to impressive speedup for large simulations. For simulations with up to 64^3 particles, we get a speedup of a factor of four.

4.2 Errors in the Modified TreePM Force

We now study errors in force for the modified TreePM force. Errors are calculated with respect to a reference force computed with very conservative values of TreePM parameters: $\theta_c = 0.01$, $r_s = 4.0$,

$r_{\rm cut} = 5.2\, r_s$. With these values, the reference force is accurate to 0.1% (Bagla & Ray 2003).

$$\epsilon = \frac{|\boldsymbol{f}_{\rm ref} - \boldsymbol{f}|}{|\boldsymbol{f}_{\rm ref}|}\,, \tag{11}$$

where ϵ, $\boldsymbol{f}_{\rm ref}$, and $\boldsymbol{f}$ are the relative error, reference force and the typical force in a simulation, respectively. We calculate errors for two distributions of particles:

- A uniform (unclustered) distribution.
- A clustered distribution taken from an $N-$body simulation.

Both distributions have $N_{\rm box}^3 = N = 128^3$ particles. The exercise we follow is similar to Bagla & Ray (2003) but now we wish to highlight the effect of groups on errors in force.

Figure 3 shows the distribution of errors for different values of θ_c. The results are shown for both kinds of distributions being studied here: the unclustered distribution (left panels) and the clustered distribution (right panels). The top row is for $c_{\rm max} = 2.0$ and the lower row is for $c_{\rm max} = 4.0$. We used $r_s = 1.0$ and $r_{\rm cut} = 5.2\, r_s$ for this figure. In the case of the unclustered distribution, the error decreases with θ_c but saturates at $\theta_c = 0.3$ and does not decrease as θ_c is decreased further. The situation is different for the clustered distribution where the errors are not sensitive to θ_c. This suggests that the errors are dominated by the long-range force. The unclustered distribution has larger errors than the clustered distribution. This is because the net force on each particle in the unclustered distribution is small, whereas the force due to a cell with many particles is large and many such large contributions have to cancel out to give a small net force. Numerical errors in adding and subtracting these large numbers seem to systematically give a large net error. Larger cells contribute for larger θ_c, hence the variation with θ_c is more dramatic in the unclustered case. This effect is apparent in the discussion of the short-range force. With $\theta_c = 0.3$, 1% of particles have errors in total force greater than 4% in the unclustered case and 1.6% in the clustered case.

The effect of the modified CAC (Eq. (5)) is seen by comparing the plots of Figure 3 for the unclustered distribution. The modified CAC is more stringent for larger values of $c_{\rm max}$ and this is clearly seen in the error for $\theta_c = 0.5$. There is a lack of variation in errors with θ_c for $\theta_c < 0.5$ indicating that at this stage, the dominant contribution to errors is from the long-range force calculation. The short-range force is more accurate with a larger $c_{\rm max}$ due to two reasons:

- The modified CAC has an r dependent opening angle threshold and requires a smaller θ_c at small distances. This is likely to reduce errors.
- The number of particles in a group is larger for larger $c_{\rm max}$. As the contribution of force from these particles is computed by direct summation over pairs, the errors are negligible.

One may raise the concern that the errors in the present approach are likely to depend on the location of a particle within the group. We have checked for anisotropies in error in the force calculations in groups that may result and we do not find any noteworthy anisotropic component.

Next, in Figure 4, we look at the errors in short-range force for the same distributions (unclustered and clustered) of particles for various values of θ_c. The reference short-range force was computed with $\theta_c = 0.01$, $r_s = 1.0$, $r_{\rm cut} = 5.2\, r_s$ and $c_{\rm max} = 4.0$. We only varied θ_c and continued to use $r_s = 1.0$, $r_{\rm cut} = 5.2\, r_s$, and $c_{\rm max} = 4.0$ for computing the short-range force and then the errors. For the purpose of computing errors in the short-range force, we cannot vary r_s between the reference and the test model.

The effect of decreasing θ_c is more dramatic on errors in the short-range force. For the unclustered case, the errors for 1% of the particles decrease by nearly 2.5 decades from $6 \times 10^{-2}\%$ to $2 \times 10^{-4}\%$ for $\theta_c = 0.1$. In the clustered case, the errors for 1% of the particles decrease by nearly 1.5 decades from $10^{-1}\%$ to $3.4 \times 10^{-2}\%$ for $\theta_c = 0.1$. One can obtain very high accuracy in short-range force by taking $\theta_c = 0.2$. As the short-range force is the dominant one at small scales, the TreePM code can be used to follow the local dynamics fairly well by using a smaller θ_c. The impact of a small θ_c on CPU time, however, remains to be seen.

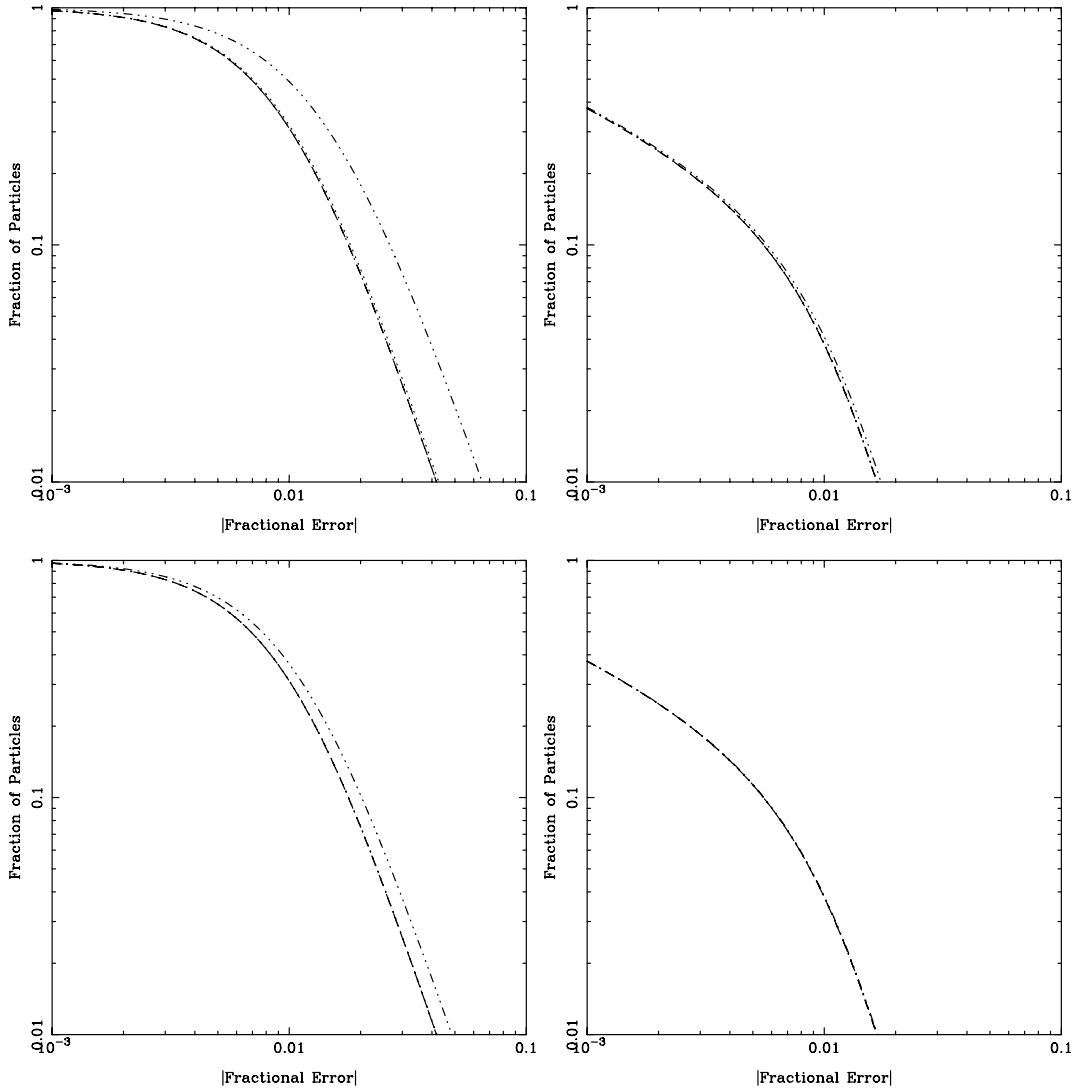

Fig. 3 Distribution of errors in total force for different values of θ_c with $c_{\max} = 2.0$ for unclustered (*top left*) and clustered (*top right*) distributions. Dashed, dot-dash-dot-dashed, dotted, dash-dot-dot-dotted lines are for $\theta_c = 0.1$, 0.2, 0.3 and 0.5 respectively. We used $r_s = 1.0, r_{\rm cut} = 5.2\, r_s$ for these plots. The corresponding plots for $c_{\max} = 4.0$ are shown in the lower left and lower right panels.

In Figure 5, we look at how the CPU time for force calculation scales with θ_c for the TreePM (left panel) and the modified TreePM (right panel). We compute the time taken for short-range force calculation per particle per timestep. We have seen in Figure 1 and the corresponding discussion that clustering does not seriously affect the performance of the TreePM code. We therefore do not repeat the exercise for distributions with different levels of clustering. We performed the short-range force timing on a clustered distribution taken from an $N-$body simulation with $N_{\rm box}^3 = N = 128^3$. We used $r_s = 1.0$ and $r_{\rm cut} = 4.0$ for the TreePM and the modified TreePM. In addition, $c_{\max} = 4.0$ and $n_{\rm pmax} = 1024$ were used for timing the modified TreePM. When θ_c is decreased from 0.5 to 0.2, the time for force computation per particle increases by 7.2% for the TreePM and 21% for the modified

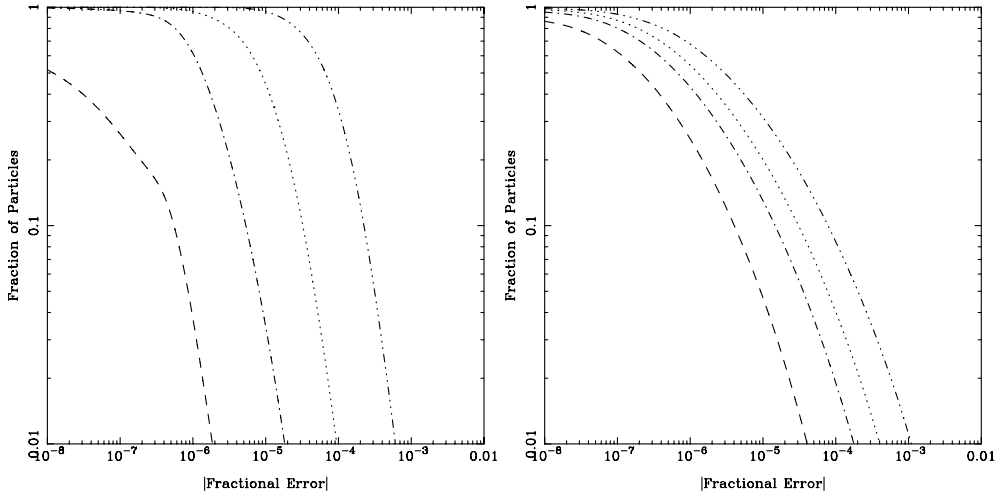

Fig. 4 Distribution of errors in short-range force for different values of θ_c for unclustered (*left panel*) and clustered (*right panel*) distributions. Dashed, dot-dash-dot-dashed, dotted, and dash-dot-dot-dotted lines are for $\theta_c = 0.1$, 0.2, 0.3 and 0.5, respectively. $c_{\rm max} = 4.0$, $r_s = 1.0$, and $r_{\rm cut} = 5.2\,r_s$ were used for these plots.

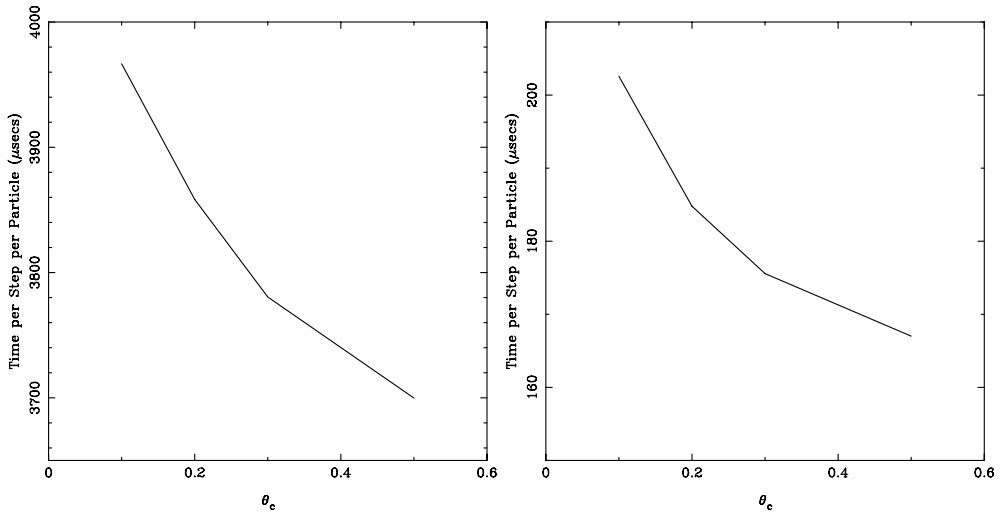

Fig. 5 Scaling of the time taken for short-range force calculation with θ_c for the TreePM (*left panel*) and the modified TreePM (*right panel*).

TreePM. The speedup of the modified TreePM over the TreePM when θ_c is reduced from 0.5 to 0.2 decreases from 22.2 to 19.6, respectively. A nice feature of TreePM codes is that unlike tree codes, the CPU time taken by TreePM codes is less sensitive to θ_c[2]. Thus, one can obtain much higher accuracy for the short-range force with a TreePM code for a considerably smaller cost in terms of the CPU time.

[2] For example, the variation in CPU time for a tree code increases by about 500% for the same change in θ_c for a simulation with $N \approx 10^4$, and the increase in CPU time is larger for simulations with a larger number of particles (Hernquist 1987).

5 A HIERARCHY OF TIMESTEPS

Due to the existence of a large range of dynamical time scales in a simulation of large scale structures, computing forces for slowly moving particles at every timestep is not required. It is better to integrate the orbits of rapidly moving particles with a smaller timestep than those that move relatively slowly; this reduces the number of force calculations that are required. As force calculation is the most time consuming component of an N-body code, this results in a significant reduction of the CPU time required. We have implemented a hierarchical time integrator similar to that used in GADGET-2 (Springel 2005), in which particle trajectories are integrated with individual timesteps and synchronized with the largest timestep. As we allow the block time step[3] to vary with time, we work with the so called KDK approach (Kick-Drift-Kick) in which velocities are updated in two half steps whereas position is updated in a full step. It can be shown that with a variable time step, KDK performs better than DKD (Drift-Kick-Drift) (see the GADGET-2 paper (Springel 2005) for details.). In our implementation of the hierarchy of time steps, the smaller time steps differ by an integer power (n) of 2 from the largest, block time step. An array is then used to store the value n which determines the timestep of the particle. The code drifts all the particles with the smallest timestep to the next time, where a force computation is done for particles that require a velocity update (Kick). We have tested the robustness of the hierarchical KDK integrator by successfully integrating the $3-$body problem discussed by Szebehely & Peters (1967).

Solving the equation of motion with a hierarchy of time steps can be combined with the group scheme. Since tree construction takes a small fraction of the total time, a new tree can be constructed whenever particles require a velocity update. The groups that contain such particles can then be identified and particles within each group can be reordered into two disjoint sets: ones that need velocity updates and others that do not. Force is computed only for particles in the first set. Since each group represents a very small fraction of the total number of particles, the overhead for reordering the particles is negligible.

Table 2 lists the time taken for a complete simulation run for the unoptimized TreePM, TreePM with hierarchical time steps, TreePM with the group scheme, and finally the TreePM with the group scheme as well as the hierarchical time steps and their speedup with respect to the base TreePM. The model used for this comparison is a power law model with $n = -1.0$ and $N_{\rm box}^3 = N = 64^3$. We used $r_s = 1.0$, $r_{\rm cut} = 5.2\,r_s$, $\theta_c = 0.5$ and $\epsilon = 0.2$ in all the runs. Here ϵ is the softening length. We used $c_{\rm max} = 4.0$ and $n_{\rm pmax} = 1024$ for the modified TreePM.

Table 2 Time taken for a complete simulation run for the unoptimized TreePM, TreePM with hierarchical time steps, TreePM with the group scheme, and finally the TreePM with the group scheme as well as the hierarchical time steps and their speedup with respect to the base TreePM.

Run	Groups	Individual Timesteps	Time (s)	Speedup w.r.t Run 1
1	No	No	401983	1.0
2	No	Yes	145240	2.77
3	Yes	No	67639	5.94
4	Yes	Yes	31612	12.72

We note that the hierarchical integrator gives a speedup of better than a factor 2, irrespective of whether the scheme of groups is used or not. The speedup is larger if the softening ϵ is smaller, as the number of levels in the hierarchy increases with decreasing ϵ. The scheme of groups on the other hand gives a speedup of 4 or better for small simulations, and a much larger speedup for bigger simulations. This speedup has little dependence on the TreePM parameters, i.e. θ_c, r_s, and $r_{\rm cut}$. The combination of the two optimizations gives us a speedup of 10 or more even for small simulations.

[3] Same as the largest time step.

6 DISCUSSION

The scheme of groups when combined with a hierarchical integrator for the equation of motion guarantees a speedup of better than 10 for any $N-$body code which uses tree structures for computing forces. From an algorithmic point of view, one does not expect a much larger speedup for larger N. However, as seen in Figure 2, the scheme also allows us to make better use of the cache on CPUs and the effective speedup can be even more impressive. We have demonstrated that memory overhead is negligible, and as was observed in Barnes (1990), this optimization just takes around 200 extra lines of code. A welcome feature is more accurate force computation than the code without this modification. This modification, in principle, introduces two additional parameters ($c_{\rm max}$, $n_{\rm pmax}$), but these are not independent and we have found that $c_{\rm max} \sim r_{\rm cut}$ and $n_{\rm pmax} \geq 10^3$ are good choices across a range of simulation sizes.

Our analysis of the optimization has been restricted to fixed resolution simulations. In the case of zoom-in simulations, the range of time scales is much larger and a more complex approach for combining the group scheme with the hierarchy of time steps may be required. The relative efficacy of the two optimizations may be very different in such a case when compared with the example studied in the previous section.

In summary, we would like to point out that the scheme of groups leads to a significant optimization of the TreePM method. The amount of CPU time saved is significant even for small simulations, but the cache optimization aspect leads to even more significant gains for large simulations. We have shown in this paper that it is possible to incorporate the scheme in a simple manner in any tree based code. The overall gain is very impressive as we are able to combine this with the use of a hierarchy of time steps. The possibility of combining the two optimizations has been explored in this work for the first time.

Acknowledgements Numerical experiments for this study were carried out at the cluster computing facility in the Harish-Chandra Research Institute (*http://cluster.hri.res.in*). This research has made use of NASA's Astrophysics Data System. The authors would like to thank Hugh Couchman, Jun Makino and Volker Springel for useful comments.

References

Bagla, J. S. 2005, Current Science, 88, 1088
Bagla, J. S. 2002, Journal of Astrophysics and Astronomy, 23, 185
Bagla, J. S., & Padmanabhan, T. 1997, Pramana, 49, 161
Bagla, J. S., & Padmanabhan, T. 1994, MNRAS, 266, 227
Bagla, J. S., & Ray, S. 2003, New Astronomy, 8, 665
Barnes, J., & Hut, P. 1986, Nature, 324, 446
Barnes, J. E. 1990, Journal of Computational Physics, 87, 161
Bernardeau, F., Colombi, S., Gaztañaga, E., & Scoccimarro, R. 2002, Phys. Rep., 367, 1
Bertschinger, E. 1998, ARA&A, 36, 599
Bode, P., & Ostriker, J. P. 2003, ApJS, 145, 1
Bode, P., Ostriker, J. P., & Xu, G. 2000, ApJS, 128, 561
Bouchet, F. R., Adam, J.-C., & Pellat, R. 1985, A&A, 144, 413
Bouchet, F. R., & Kandrup, H. E. 1985, ApJ, 299, 1
Bouchet, F. R., & Hernquist, L. 1988, ApJS, 68, 521
Brainerd, T. G., Scherrer, R. J., & Villumsen, J. V. 1993, ApJ, 418, 570
Brieu, P. P., & Evrard, A. E. 2000, New Astronomy, 5, 163
Brieu, P. P., Summers, F. J., & Ostriker, J. P. 1995, ApJ, 453, 566
Couchman, H. M. P. 1991, ApJ, 368, L23
Davis, M., & Peebles, P. J. E. 1977, ApJS, 34, 425
Dehnen, W. 2002, Journal of Computational Physics, 179, 27
Dehnen, W. 2000, ApJ, 536, L39
Dubinski, J. 1996, New Astronomy, 1, 133
Dubinski, J., Kim, J., Park, C., & Humble, R. 2004, New Astronomy, 9, 111
Ebisuzaki, T., Makino, J., Fukushige, T., Taiji, M., Sugimoto, D., Ito, T., & Okumura, S. K. 1993, PASJ, 45, 269

Efstathiou, G., Davis, M., White, S. D. M., & Frenk, C. S. 1985, ApJS, 57, 241
Efstathiou, G., Frenk, C. S., White, S. D. M., & Davis, M. 1988, MNRAS, 235, 715
Greengard, L., & Rokhlin, V. 1987, Journal of Computational Physics, 73, 325
Gurbatov, S. N., Saichev, A. I., & Shandarin, S. F. 1989, MNRAS, 236, 385
Hamilton, A. J. S., Kumar, P., Lu, E., & Matthews, A. 1991, ApJ, 374, L1
Hernquist, L. 1990, Journal of Computational Physics, 87, 137
Hernquist, L. 1987, ApJS, 64, 715
Hernquist, L., Bouchet, F. R., & Suto, Y. 1991, ApJS, 75, 231
Hockney, R. W., & Eastwood, J. W. 1988, Computer Simulation using Particles, McGraw-Hill
Hui, L., & Bertschinger, E. 1996, ApJ, 471, 1
Jain, B., Mo, H. J., & White, S. D. M. 1995, MNRAS, 276, L25
Jernigan, J. G., & Porter, D. H. 1989, ApJS, 71, 871
Kanekar, N. 2000, ApJ, 531, 17
Kawai, A., & Makino, J. 2001, ApJ, 550, L143
Klypin, A. A., & Shandarin, S. F. 1983, MNRAS, 204, 891
Knebe, A., Green, A., & Binney, J. 2001, MNRAS, 325, 845
Kravtsov, A. V., Klypin, A. A., & Khokhlov, A. M. 1997, ApJS, 111, 73
Ma, C.-P. 1998, ApJ, 508, L5
Macfarland, T., Couchman, H. M. P., Pearce, F. R., & Pichlmeier, J. 1998, New Astronomy, 3, 687
Makino, J. 2004, PASJ, 56, 521
Makino, J. 2002, New Astronomy, 7, 373
Makino, J. 1991, PASJ, 43, 621
Makino, J. 1990, Journal of Computational Physics, 87, 148
Makino, J., Fukushige, T., Koga, M., & Namura, K. 2003, PASJ, 55, 1163
Matarrese, S., Lucchin, F., Moscardini, L., & Saez, D. 1992, MNRAS, 259, 437
Merz, H., Pen, U.-L., & Trac, H. 2005, New Astronomy, 10, 393
Miller, R. H. 1983, ApJ, 270, 390
Nityananda, R., & Padmanabhan, T. 1994, MNRAS, 271, 976
Padmanabhan, T. 2002, Theoretical Astrophysics, Volume III: Galaxies and Cosmology (Cambridge University Press)
Padmanabhan, T. 1996, MNRAS, 278, L29
Padmanabhan, T., Cen, R., Ostriker, J. P., & Summers, F. J. 1996, ApJ, 466, 604
Peacock, J. A., & Dodds, S. J. 1996, MNRAS, 280, L19
Peacock, J. A., & Dodds, S. J. 1994, MNRAS, 267, 1020
Peacock, J. A. 1999, Cosmological Physics (Cambridge University Press)
Peebles, P. J. E. 1980, The Large-Scale Structure of the Universe (Princeton University Press)
Peebles, P. J. E. 1974, A&A, 32, 391
Percival, W. J., et al. 2007, ApJ, 657, 645
Ray, S., & Bagla, J. S. 2004, arXiv:astro-ph/0405220
Sahni, V., & Coles, P. 1995, PhR, 262, 1
Salmon, J. K., & Warren, M. S. 1994, Journal of Computational Physics, 111, 136
Smith, R. E., et al. 2003, MNRAS, 341, 1311
Spergel, D. N., et al. 2007, ApJS, 170, 377
Springel, V., Yoshida, N., & White, S. D. M. 2001, New Astronomy, 6, 79
Springel, V. 2005, MNRAS, 364, 1105
Springel, V., et al. 2005, Natur, 435, 629
Suisalu, I., & Saar, E. 1995, MNRAS, 274, 287
Szebehely, V., & Peters, C. F. 1967, AJ, 72, 1187
Thacker, R. J., & Couchman, H. M. P. 2006, Computer Physics Communications, 174, 540
Theuns, T. 1994, Computer Physics Communications, 78, 238
Wadsley, J. W., Stadel, J., & Quinn, T. 2004, New Astronomy, 9, 137
Xu, G. 1995, ApJS, 98, 355
Yoshikawa, K., & Fukushige, T. 2005, PASJ, 57, 849
Zel'Dovich, Y. B. 1970, A&A, 5, 84