UNIVERSITY OF AMSTERDAM

UvA-DARE (Digital Academic Repository)

Hybrid Systems for N-body Simulations

Spinnato, P.F.

Publication date
2003

Link to publication

Citation for published version (APA):
Spinnato, P. F. (2003). *Hybrid Systems for N-body Simulations*. [Thesis, fully internal, Universiteit van Amsterdam]. Eigen Beheer.

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, P.O. Box 19185, 1000 GD Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (https://dare.uva.nl)

Download date:12 Feb 2026

Chapter 4

Pseudo-Particle Powered Treecode: Error Analysis and Optimisation[†]

In this chapter we study the pseudo-particle scheme, which makes it possible to model higher order multipole moments for the treecode on the GRAPE. The treecode, introduced in section 1.4.2, offers excellent scaling for the simulation of self-gravitating systems, but at the cost of limited accuracy. The pseudo-particle approach, where a multipole expansion is expressed in terms of a particle distribution, provides an accuracy that it is easy to tune, and is suitable for making full use of the ultra fast GRAPE Special Purpose Device for the gravity force computation. The GRAPE is introduced in section 1.3 and extensively studied in part I of this thesis. We study the error behaviour of this approach, comparing it with the standard treecode, and introduce improvements that reduce the errors. Furthermore we present an extension of the pseudo-particle scheme, where pseudo-particles are not fixed in space, but move following the physical particle distribution. This extension decreases the computational overhead due to pseudo-particle recomputation, and optimises the scheme for the use on GRAPE and on parallel systems.

4.1 Introduction

The treecode (Barnes & Hut, 1986), introduced in section 1.4.2, is one of the most popular numerical methods for particle simulation involving long range interactions in astrophysical contexts. Its operation count scales as $\mathcal{O}(N \log N)$, which is a great improvement compared

[†]This chapter is based on work published in:

P.F. Spinnato; S.F. Portegies Zwart; M. Fellhauer; G.D. van Albada and P.M.A. Sloot: *Tools and Techniques for N-body Simulations*, in R. Capuzzo Dolcetta, editor, Proceedings of the 1st workshop on Computational Astrophysics in Italy: Methods and Tools, MemSAIt Suppl. Series vol. 1, pp. 54–65. Società Astronomica Italiana, 2003.

P.F. Spinnato; G.D. van Albada and P.M.A. Sloot: *Pseudo-Particle Powered Treecode: Error Analysis and Optimisation*, to be submitted to Journal of Computational Physics, 2003.

with the $\mathcal{O}(N^2)$ scaling of the direct particle-particle method, where the force on a particle is computed by directly evaluating the contributions of all other particles (see section 1.4.1 and 2.2.2). The treecode speedup is obtained at the cost of a reduced accuracy. The treecode groups particles together according to a tree data structure, where each node of the tree is associated with a cubical cell in three-dimensional space, and a cell corresponding to a given node consists of the eight cells corresponding to the eight subnodes hierarchically connected to the given node. The treecode attains its $\mathcal{O}(N \log N)$ scaling by computing force on a particle i from larger and larger cells as their distance from i gets bigger and bigger. In order to decide whether a cell is far enough to be accepted for such force computation, a suitable Multipole Acceptability Criterion (MAC) must be provided. In section 4.3 below MACs are discussed further.

The force contribution is evaluated from a multipole expansion of the particle distribution contained in the cell. The accuracy of the evaluation depends on the highest term in the multipole expansion, on the MAC used, and on the actual value chosen for the MAC parameter. Usually, expansions are truncated at the quadrupole term, leading to an accuracy in the force in the order of 1% (Salmon & Warren, 1994) for commonly used MAC settings. This makes treecodes unsuitable for applications that require a high numerical accuracy. Better accuracies can be obtained by using different MAC settings, but this will greatly increase the computing cost.

Research has been carried out in order to improve the treecode accuracy, by increasing the maximal multipole expansion order (McMillan & Aarseth, 1993). We aim at developing a version of the treecode that allows a tunable accuracy, while limiting the impact on code performance. We design our code in order to be optimally run on a parallel platform that includes the GRAPE boards (Makino & Taiji, 1998). GRAPE, as described in section 1.3, is a special purpose device implementing an array of fully hardwired pipelines, each one computing the gravitational interaction between two particles in a single clock cycle.

Our code derives from the pseudo-particle approach proposed by Makino (1999), and implemented by Kawai & Makino (2001), developing an early idea of Anderson (1992). Makino and Kawai implemented a serial pseudo-particle treecode that makes use of GRAPE (Kawai, 1999; Kawai & Makino, 2001). The algorithm that they propose places the pseudo-particles in fixed positions on a spherical surface surrounding the physical distribution, and computes the pseudo-particle masses as weighted sums of the physical particle masses (see eq. (4.1) below). The pseudo-particle approach allows the treecode to take advantage of both the very high computing speed offered by the GRAPE, and makes it very easy to increase the code accuracy by increasing the maximal multipole order. In the standard treecode, multipoles are computed in terms of series expansions (see, e.g., McMillan & Aarseth, 1993). Mathematical expressions of the higher moment terms are increasingly cumbersome, and not easy to implement. Conversely, in the pseudo-particle scheme, increasing the maximal multipole order of the expansion is simply a matter of increasing the number of pseudo-particles that make up the expansion.

When one runs a code using multipole expansions, such as the treecode, with the GRAPE, a fundamental problem arises. GRAPE can only compute particle-particle interactions, hence the advantage of lumping particles to obtain a single multipole expansion is

wasted: in the standard treecode formulation, such expansions are expressed in terms of spatial derivatives of $1/r^2$, consequently GRAPE cannot compute particle-multipole interactions. This implies that all interactions involving a multipole term must be evaluated by the host computer. The key idea of the pseudo-particle approach is to use the Anderson (1992) formulation for the multipole expansion which, instead of a complicated polynomial, is given in terms of a pseudo-particle distribution. In this way, GRAPE is able to compute also the force contribution from higher multipole terms, since they are now expressed in terms of a particle distribution.

In order to have a quantitative estimate of the accuracy in the pseudo-particle scheme, we have carried out an error analysis study, comparing the accuracy of the pseudo-particle code with two implementations of the standard treecode. We compare our code with the code of Salmon & Warren (1994), and with GADGET (Springel *et al.*, 2001). We introduce a modification in the method that improves its accuracy when the physical particle distribution is highly inhomogeneous. Moreover, we modify the Makino and Kawai pseudo-particle method by introducing pseudo-particle position extrapolation, in order to decrease the overhead due to pseudo-particle re-evaluation. Instead of recomputing the pseudo-particle expansion at each iteration, we extrapolate the pseudo-particle positions for a number of time-steps; in order to accomplish this, we define a pseudo-particle velocity. Such scheme is suited for use on parallel systems hosting GRAPE boards, as discussed in section 3.4.4. We first present the pseudo-particle method, and discuss the improvement that we introduce. We continue with the error evaluation, then present the moving pseudo-particle scheme, and finally discuss our results and future work.

4.2 The pseudo-particle method

The pseudo-particle method approximates the multipole expansion of a given set of particles by means of a pseudo-particle distribution. The pseudo-particle positions are fixed, and lie on a spherical surface surrounding the real particle distribution (Makino, 1999; Kawai, 1999). Optimised distributions, with minimal number of pseudo-particles, have been obtained up to the quadrupole moment (Kawai, 1999; Kawai & Makino, 2001). The mass of each pseudo-particle is given by:

$$M_k = \frac{1}{K} \sum_{1}^{N}{}_j m_j \sum_{0}^{p}{}_l \left(\frac{r_j}{a}\right)^l (2l+1)\, P_l(\cos\gamma_{jk})\,, \tag{4.1}$$

where M_k is the mass of pseudo-particle k, K is the total number of pseudo-particles, N is the number of real particles from which the pseudo-particle expansion is computed, m_j is the mass of the real particle j, p is the maximal order of the multipole expansion, r_j is the norm of the position vector of particle j, a is the radius of the pseudo-particle sphere, P_l is the modified Legendre polynomial of order l, and finally γ_{jk} is the angle between the position vectors of particle j and pseudo-particle k.

As will be shown later, the pseudo-particle approximation based on a spherical distribution suffers from limitations in representing non-uniform mass distributions. In order

to solve this problem, we introduce a simple modification, that substantially improves the accuracy of the method. It consists in retaining the spherical pseudo-particle distribution, with pseudo-particle masses that now represent only the higher multipole moments of the real particle distribution $\mathcal{R}$, starting from the quadrupole moment. An extra pseudo-particle is added at the centre of mass of $\mathcal{R}$, with mass equal to the total mass of $\mathcal{R}$. The extra pseudo-particle accounts for the monopole and dipole moment, and improves the ability of the pseudo-particle distribution to represent non-uniform real distributions.

The higher order expansion of $\mathcal{R}$, i.e. the part that does not contains the monopole and dipole terms, is obtained with a simple expedient. It consists in computing the pseudo-particle masses still using equation (4.1), but adding to the N particles of $\mathcal{R}$ a "virtual" particle which has the effect of removing the monopole and dipole moments from the expansion of $\mathcal{R}$. This virtual particle is placed at the centre of mass of $\mathcal{R}$, and has a mass equal to the opposite of the total mass of $\mathcal{R}$. The pseudo-particle distribution resulting from this combination of the N real particles and the negative mass particle accounts for the higher order expansion of $\mathcal{R}$. Finally, we obtain the complete multipole expansion by adding to this pseudo-particle distribution the extra pseudo-particle located at the centre of mass of $\mathcal{R}$.

4.3 Error evaluation

4.3.1 Comparisons

First we validate our implementation of the unmodified pseudo-particle method against the implementation of Kawai and Makino. In Kawai & Makino (1999) the error on the potential generated by pseudo-particle expansions up to a given multipole order p is presented. The real particle distribution consists of a single particle i of unit mass placed at position $\mathbf{p} = (1, 0, 0)$ in spherical coordinates. The potential $\Phi = -1/|\mathbf{r} - \mathbf{p}|$ generated by i is computed along a straight line, at points $\mathbf{r} = (r, 2\pi/3, 0)$, for r varying within a certain range. The relative error $|(\Phi_p - \Phi)/\Phi|$, where Φ_p is the potential given by the pseudo-particle expansion up to order p, is presented in fig. 4.1 for both our implementation and the one of Kawai & Makino (1999).

Our values, labelled "PP", are in very good agreement with Kawai and Makino's values, labelled "KM". Irregularities in the error profiles lead to local differences, arguably due to differences in the exact positions of the pseudo-particles between the two implementations, which result in local differences in the value of the potential. The global trend is however very similar in the two cases.

In the next section we will study the worst-case error behaviour of our implementation, comparing our results with a similar analysis carried out in Salmon & Warren (1994).

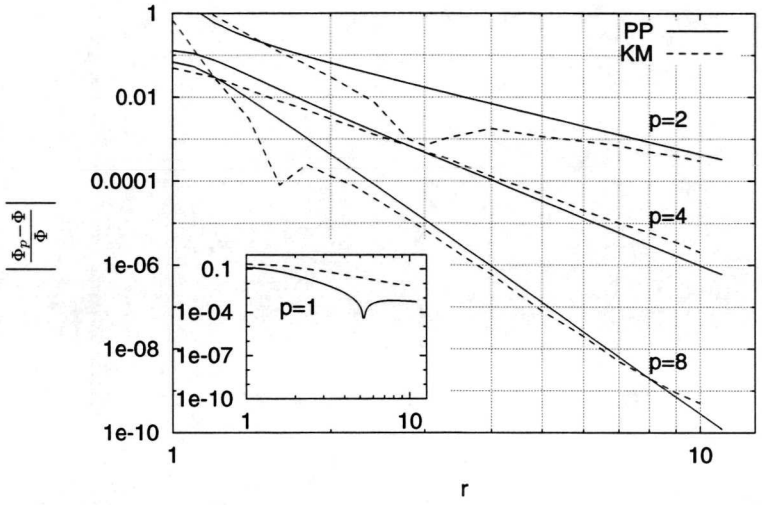

Figure 4.1: Relative error in the evaluation of the potential generated by a unit mass particle placed at position $\mathbf{p} = (1, 0, 0)$ in spherical coordinates. The error is measured at position $\mathbf{r} = (r, 2\pi/3, 0)$, and plotted as a function of r. Values from our implementation are plotted with solid lines, and labelled as PP. Values from Kawai & Makino (1999) are plotted with dashed lines, and labelled as KM. p is the maximal multipole expansion order. The pseudo-particle sphere radius is $a = 1$.

4.3.2 Worst-case error

It is important to analyse the behaviour of the method in the worst case configuration, i.e. the situation leading to the highest error in a force evaluation, even though this situation is unlikely to arise in actual simulations. This analysis gives upper bounds to the code error, and provides a good test for comparative analysis of different multipole acceptability criteria. In order to perform the worst-case analysis of our code, we follow the same procedure as Salmon & Warren (1994) (referred as SW hereafter). The worst-case configuration consists of two point particles placed at two opposite corners of a cubic cell.

Preliminary tests showed that the highest error occurs when the mass m_1 of the particle closer to the evaluation point is much lower than the other particle mass. We set then $m_1 = 10^{-5}$ and $m_2 = 1 - m_1$. We compute the gravitational acceleration exerted by the two particles along a straight line overlapping with the diagonal of the cell where the particles are located. We compute both the exact acceleration $\mathbf{a}$, and the acceleration given by the pseudo-particle expansion up to a given multipole, $\mathbf{a}_p$. From that, the error is evaluated as:

$$\epsilon_p = \frac{|\mathbf{a} - \mathbf{a}_p|}{|\mathbf{a}|}. \tag{4.2}$$

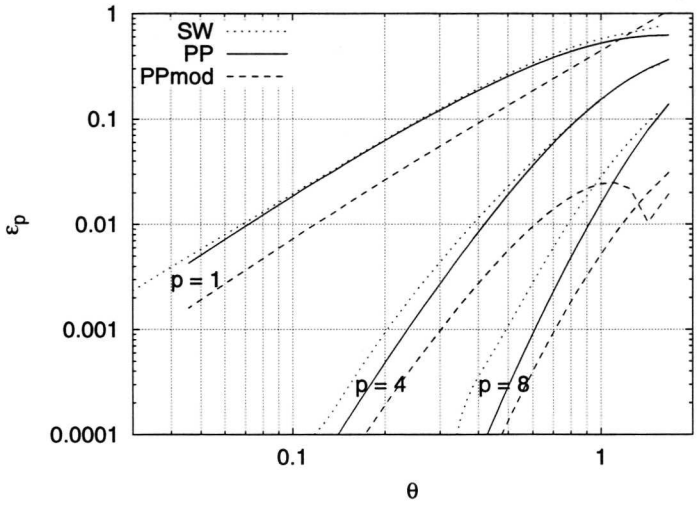

Figure 4.2: Comparison of the worst-case errors in the acceleration. Data are plot as a function of the opening parameter θ. Results from our implementations are labelled PP for the canonical pseudo-particle method, and PPmod, for the modified method where an extra pseudo-particle is added. The Salmon & Warren (1994) results are labelled SW; p is the maximal multipole expansion order.

The Multipole Acceptability Criterion (MAC) we adopt is the Minimal Distance (MD) criterion (Salmon & Warren, 1994). According to the MD MAC, a multipole expansion is accepted if

$$\frac{l}{d} < \theta \tag{4.3}$$

where l is the cell size, d is the minimal distance of the evaluation point from the cell, and θ is an input parameter, usually $\theta \lesssim 1$. The original Barnes & Hut (1986) MAC differs from the MD MAC in the definition of d. Barnes and Hut define d as the distance of the evaluation point from the centre of mass of the cell.

We evaluated the error defined in eq. (4.2) for $p \in \{1, 2, 4, 8\}$, and compared our results with the results presented by SW, fig. 5. The results are shown in fig. 4.2. Data are plotted as a function of the opening parameter θ, in order to show what is the largest error to be expected for a given value of θ. We plot our results, labelled "PP", and SW's results, labelled "SW". Data for the case $p = 2$ are omitted for the sake of readability. We also plot the error of our modified pseudo-particle method, described above. Results from this method are labelled "PPmod". Since our modified method adapts very well to highly non-uniform distributions, the distribution used for the PP method is not the worst case for the PPmod method. Numerical tests showed that the worst case is now when m_1 is about one order of magnitude smaller than m_2, with very little dependence of the errors on the precise value of the masses. We thus chose $m_1 = 0.1$. In all cases, the cell size is $l = 1$, and the

pseudo-particle sphere radius is $a = 1$.

The PP curves are in very good agreement with the SW curves, with a tendency for the PP curves to have smaller errors for lower values of θ in the high precision cases ($p \in \{4, 8\}$). The agreement of our results with the MD case of Salmon and Warren is not surprising, since we adopt the same criterion and the same geometry for the error analysis. The PPmod results are always better than the PP and SW cases, especially for the low precision cases. The improvement obtained with the PPmod method will be also observed in the statistical error analysis.

4.3.3 Statistical error

We compare the statistical error of the pseudo-particle code with the standard treecode results presented in Salmon & Warren (1994). We use this implementation as our benchmark, since multipole terms are computed there with the standard method, i.e. by means of series expansions. Specifically, we compare our results with the isolated halo case of SW. In that experiment, 4942 particles were chosen at random from a high density core distribution, and the error analysis was performed on them, using eq. (4.2) for the error estimation. Our results, obtained using the MD MAC (in)eq. (4.3), and opening angle $\theta = 1.1$, are compared with the same case presented in Salmon & Warren (1994), fig. 11. The configuration that we used includes 4096 particles. In our implementation of the pseudo-particle treecode, each non-terminal cell is associated with a pseudo-particle distribution located on the surface of a sphere whose radius is one half of the cell size. A sphere radius smaller than the cell size gives a better accuracy (Kawai, 1999), and preliminary tests gave us one half of the cell size as the optimal value for the sphere radius. The pseudo-particle distribution of a parent cell is obtained recursively from the distributions of its child cells. A particle-cell interaction now becomes a set of particle-particle interactions between the particle and the pseudo-particle distribution of that cell. The multipole expansion is computed up to the quadrupole moment.

The error distribution of the pseudo-particle scheme is shown in fig. 4.3. We computed the relative error (eq. (4.2)) for the force on each particle, then obtained the cumulative percentile distribution shown in the figure. This method of analysing the error gives much more insight into the accuracy of the code, than for instance rms or maximal error. An optimal code has a flat error distribution, so that the great majority of errors have about the same value. A code with a wide spread in error values leads to a waste of compute time, since increasing the accuracy in order to reduce large errors, results in unnecessary refinement for those force computations whose error was already small. See Salmon & Warren (1994) for a more extensive discussion.

The "PP" case in fig. 4.3 shows how much larger are the errors in the unmodified PP code with respect to the standard code. These tests are performed on a dark halo distribution (Hernquist, 1990), which is a highly inhomogeneous particle distribution, having a radial density $\rho(r) \propto [r \cdot (1 + r^3)]^{-1}$.[1] A dark halo is the result of the gravitational collapse

[1]Nowadays $\rho(r) \propto [r \cdot (1 + r)^2]^{-1}$ (Navarro *et al.*, 1997) is the most accepted density profile. We chose to use the other profile, to be consistent with the profile used by Salmon & Warren (1994).

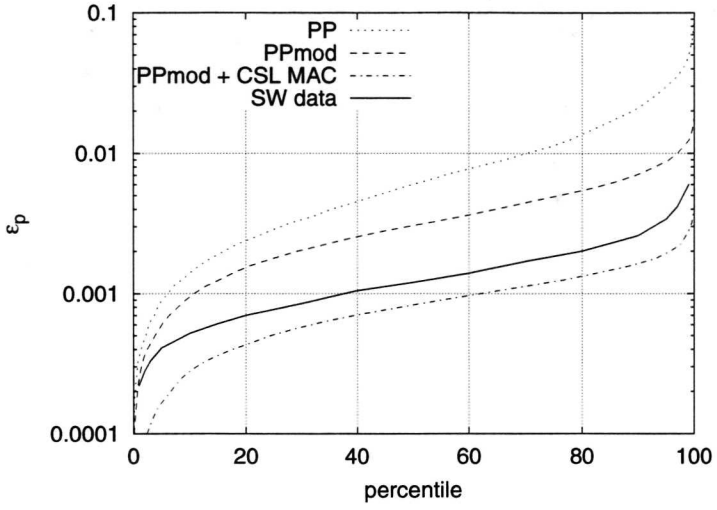

Figure 4.3: Error distribution for the PP code, compared with Salmon & Warren data (Salmon & Warren, 1994). "PP" refers to the unmodified code, "PPmod" refers to the code with the extra pseudo-particle placed at the centre of mass. "PPmod + CSL MAC" refers to the modified PP code with the MAC according to (in)eq. 4.4. The multipole expansion is up to the quadrupole moment.

of intergalactic gas due to random density peaks. Tests performed with more homogeneous particle distributions showed that the PP code accuracy is much better is those cases. The same tests, performed with the standard code, showed that the canonical treecode is less sensitive to the spatial distribution of the particles. Indeed, errors are higher for the uniform distribution, which, conversely, is the best case for the pseudo-particle code.

The pseudo-particle expansion accuracy suffers strongly from a non-uniform distribution of particles. For this reason, as already mentioned, we modified it by introducing in the multipole expansion an extra pseudo-particle located at the centre of mass of the distribution. This allows us to represent highly inhomogeneous distributions much better. The "PPmod" case in fig. 4.3 shows the error behaviour of our modified pseudo-particle treecode. The errors are now much smaller than the ones for the unmodified code. Yet, errors are still higher than the ones of the standard treecode.

We observed that most of the large errors come from distant cells. In order to control this error, we modified the MAC in such a way that a multipole expansion is accepted if:

$$\frac{l}{\frac{d}{\sqrt{d}+1}} < \theta \quad , \tag{4.4}$$

where symbols have the same meaning as in (in)eq. (4.3). The opening criterion in (in)eq. (4.4) reduces the acceptability of far-away (hence large) cells, at the cost of an increased computational load. This new Cell Size Limiting (CSL) criterion, applied to the modified pseudo-particle code, gives a remarkable improvement in the code accuracy, as shown in

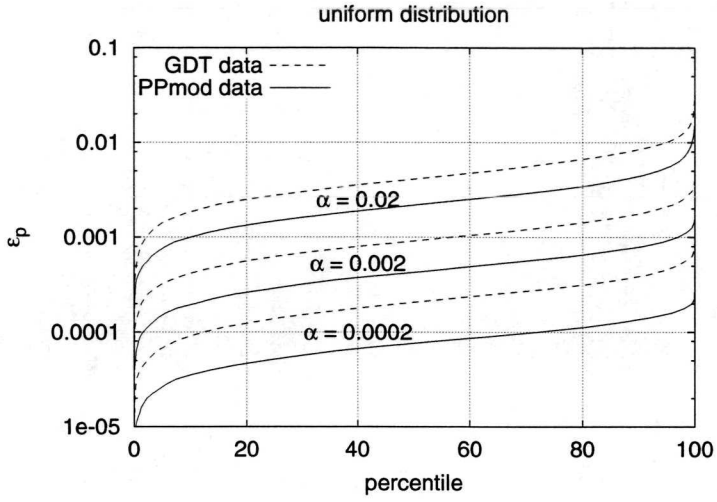

Figure 4.4: Comparison of the error distributions for our code (labelled PPmod) and the treecode GADGET (labelled GDT), where multipoles are computed in the standard way. The MAC of (in)eq. 4.5 is used here. For consistency with the standard treecode, cell-particle distances in our code are not minimal distances as in the previous cases, but are measured with respect to the cell centre of mass. The multipole expansion is up to the quadrupole moment.

the "PPmod + CSL MAC" case in fig. 4.3. Now the error of the pseudo-particle code is below the error of the standard treecode.

An extra factor that increases the pseudo-particle code accuracy is the fact that in our implementation a multipole contribution is evaluated only if there are more particles in the cell than the pseudo-particles used to represent the multipole expansion. In the present case, with expansions truncated at the quadrupole term, each expansion has 13 pseudo-particles. The pseudo-particle code accuracy benefits from this, since force from cells containing 13 particles or fewer is always computed exactly. Moreover, we also gain in performance, since fewer interactions are computed in this way. The effect of this feature of the pseudo-particle code will be studied further in next section.

4.3.4 The GADGET MAC

The criterion in (in)eq. (4.3) and the modified CSL version in (in)eq. (4.4) are based on the principle that the error from a certain cell will be small if that cell is "seen" under a small opening angle. A criterion that directly estimates the error that the cell expansion will introduce if accepted, could lead to a more efficient MAC. This approach was proposed in Salmon & Warren (1994), and is implemented in the recently developed treecode GADGET (Springel *et al.*, 2001), which we already studied in our performance modelling studies presented in chapter 3. According to the MAC discussed in Springel *et al.* (2001), a multipole expansion

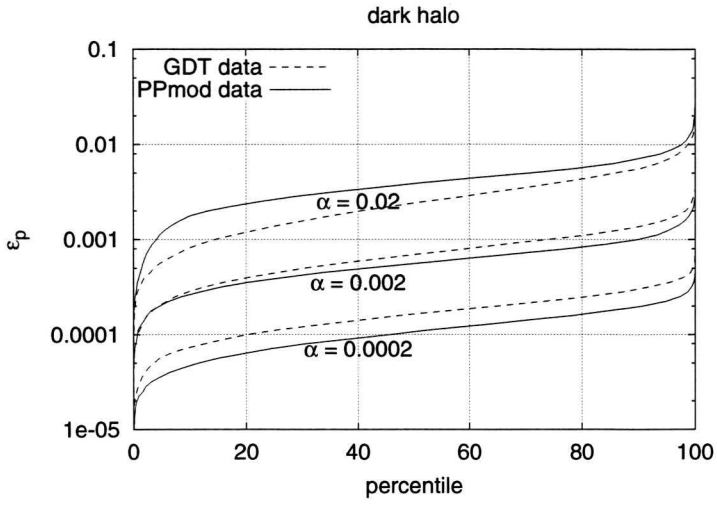

Figure 4.5: Comparison of the error distributions for a highly clustered configuration. Symbols are the same as in fig. 4.4. The multipole expansion is up to the quadrupole moment.

is accepted if:

$$\frac{Ml^4}{d^6} < \alpha |\mathbf{a}_{old}| \quad , \tag{4.5}$$

where M is the cell mass, l is the cell size, d is the particle-cell distance, α a numerical coefficient, and $\mathbf{a}_{old}$ the previous value of the acceleration on the particle currently dealt with. The left hand side of the above expression can be seen as a rough estimate of the force contribution from the hexadecapole moment of the cell (Springel *et al.*, 2001), and $\mathbf{a}_{old}$ is an estimate of the true current value of the particle acceleration. An estimate of the error introduced by truncating a multipole expansion at a certain order, is given by the contribution of the first term not included in the expansion. In the case of GADGET, the truncation is at the quadrupole term. The octupole term should then be chosen. However, the octupole term vanishes for uniform distributions, in those cases the hexadecapole term should be used. The authors chose to use the estimate of the hexadecapole term contribution in all cases. This improves accuracy, and is also cheaper to compute, because it does not involve square root evaluations. The criterion in (in)eq. (4.5), states that a cell is accepted if the estimate of the hexadecapole term contribution is less than a small fraction of the total force on the particle. If higher accuracy is required, the estimate of a higher multipole term should be used in the left hand side of (in)eq. (4.5). This MAC opens a cell only if the expected error from the cell is large. Nearby cells that would be opened with the canonical MAC (in)eq. (4.3) because they are "seen" under a large opening angle, now are not opened if their effect on the total force error is small. Because $\mathbf{a}_{old}$ is not defined in the first code iteration, the very first force evaluation is still performed using the canonical MAC.

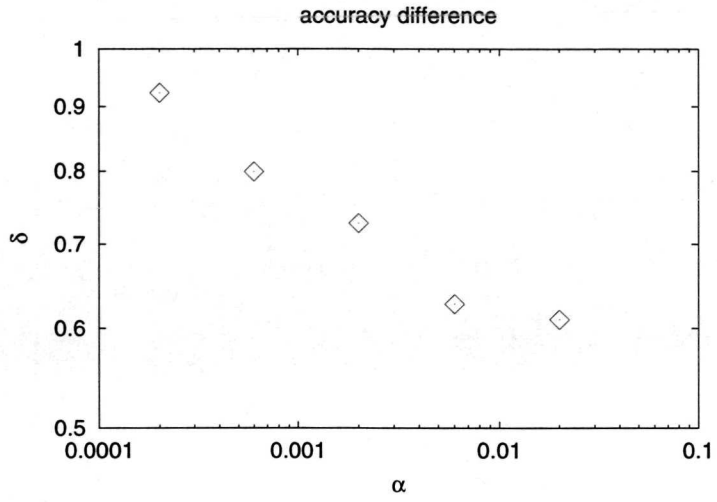

Figure 4.6: Relative difference between the error values of the two codes, measured at the 50% percentile, as a function of the accuracy parameter α.

We implemented the criterion defined in the (in)eq. 4.5, and compared our results with the standard treecode for two different particle configurations: a uniform distribution, and a highly concentrated (dark halo) distribution. Our results are presented in fig. 4.4 and 4.5 for three representative values of α. Except for the low accuracy case in the highly concentrated distribution, the pseudo-particle code shows a better accuracy than the GADGET code, especially in the uniform distribution case. This confirms the tendency of the pseudo-particle code to give better results with homogeneous distributions.

In the uniform distribution case, the relative separation

$$\delta = \frac{|\epsilon_p^{GDT} - \epsilon_p^{PP}|}{(\epsilon_p^{GDT} + \epsilon_p^{PP})/2} \tag{4.6}$$

between the results of the two codes seems to be dependent on α. In fig. 4.6 we show the values of δ, measured according to eq. (4.6) at the 50% percentile value of ϵ_p. Measures for two more values of α have been added in this case. The relative difference between the two codes tends to decrease and saturate with α. This can be explained by the fact that a smaller value of α causes a smaller size of the accepted cells. Smaller cells contain fewer particles, and if the number of particles is 13 or less (see discussion at the end of previous section), the PPmod code computes forces from the cell directly, hence with perfect accuracy. This explains why the accuracy of the PPmod code is higher for smaller values of α.

The effect of this feature of the PPmod code will be less pronounced if the number of particles is increased. In this case, fewer cells, among those that pass the MAC (in)eq. (4.5) will contain 13 particles or fewer. If the total number of particles is increased by a factor n, the number of particles in a given cell is also increased by the same factor. The fraction of

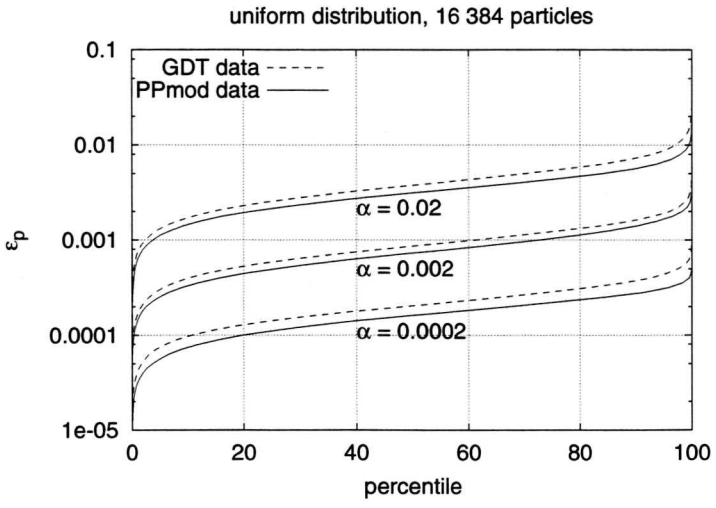

Figure 4.7: Comparison of the error distributions for a uniform distribution, and 16 384 particles. Symbols are the same as in fig. 4.4. The multipole expansion is up to the quadrupole moment.

accepted cells whose force will be computed exactly by the PPmod code will then decrease, so that the accuracy gain of the PPmod code with respect to the standard code will decrease. Fig. 4.7 shows a comparison of the PPmod code and the standard code with 16 384 particles. The separation between the results of the two codes is clearly smaller with respect to the one in fig. 4.4. For larger numbers of particles the separation between the two codes is likely to become negligible.

We used our pseudo-particle code to compare the two MACs of (in)eqs. (4.4) and (4.5), in order to show the error profile that they produce, as a function of the accuracy parameter. Our results are presented in fig. 4.8. We compare the two MACs using cases having the same accuracy, measured according to the respective accuracy parameters, i.e. the opening angle θ of the CSL MAC (see (in)eq. (4.4)), and the accuracy parameter α of the GADGET MAC (see (in)eq. (4.5)). It is clear how the GADGET MAC gives better results in terms of flatness of the error profile. The total computational load, measured in terms of the mean number of interactions per particle κ, is of the same order for cases having comparable accuracy.

4.4 Moving pseudo-particle scheme

The evaluation of the pseudo-particle masses is computationally expensive, especially when a high multipole order is required (Kawai, 1999; Makino, 1999). Moreover, when the GRAPE hardware is used, the recomputed pseudo-particle data must be reloaded at each iteration. This introduces an high overhead, limiting the convenience of the pseudo-particle approach. We propose a scheme that does not require a re-evaluation of the pseudo-particle masses

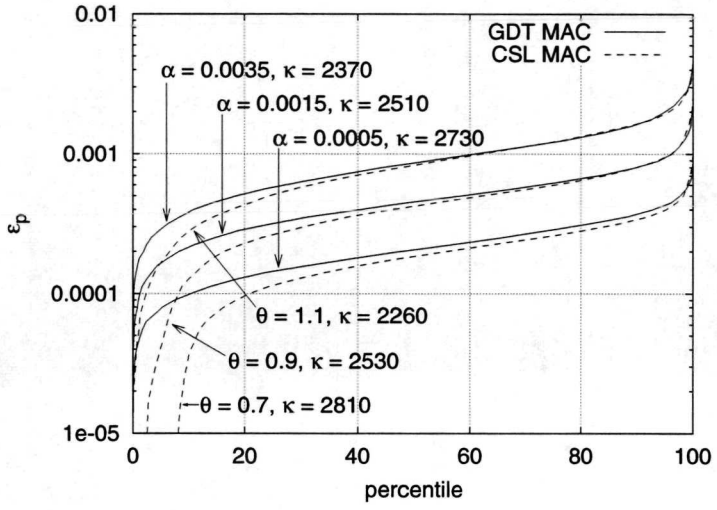

Figure 4.8: Comparison of the error profiles of the two MACs of (in)eqs. (4.4) and (4.5). θ is the opening angle of the CSL MAC of (in)eq. (4.4), α is the accuracy parameter of the GDT MAC of (in)eq. (4.5), κ is the mean number of interactions per particle.

at each iteration. Instead, we assign a velocity to the pseudo-particles, and let them move with this velocity. Velocities must be assigned so that the pseudo-particles' motion correctly reproduces the changes in the moment distribution for each cell. The advantage of this approach is that pseudo-particle data must be recomputed less frequently. Moreover, when GRAPE is used, no reload is necessary, since GRAPE contains the hardware needed to extrapolate particle positions. We compute the pseudo-particle momenta adapting the formula used to compute the pseudo-particle mass eq. (4.1):

$$\mathbf{p}_k = \frac{1}{K} \sum_{1}^{N}{}_j \, \mathbf{p}_j \sum_{0}^{P}{}_l \left(\frac{r_j}{a}\right)^l (2l+1) \, P_l(\cos\gamma_{jk}) \quad . \tag{4.7}$$

Here $\mathbf{p}_k$ is the momentum of pseudo-particle k, $\mathbf{p}_j$ is the momentum of the real particle j, all other symbols have the same meaning as in eq. (4.1).

We present below a statistical error analysis of our moving pseudo-particle scheme, similar to the analysis presented in the previous section.

4.4.1 Statistical error

In order to carry out a statistical error analysis of the moving pseudo-particle scheme, we used here a dark halo configuration. In this case, we used a configuration with 40 000 particles, in order to reduce the fraction of cells containing 13 particles or less. In this way we increase the

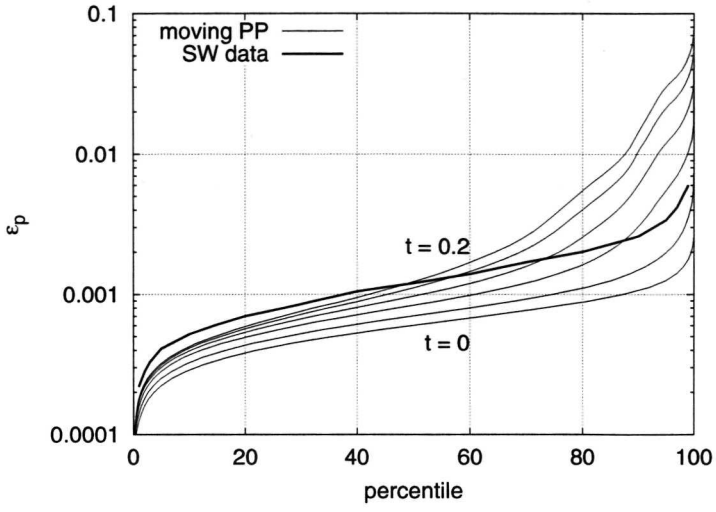

Figure 4.9: Error distribution for the moving pseudo-particle code. The error profile at various extrapolation times is compared with a canonical treecode error profile. We use the MAC defined in eq. (4.5), with $\alpha = 0.002$. The time interval between each error profile is equal to 0.04. In this case is $N = 40\,000$.

fraction of contributions from multipoles with respect to contributions from single particles. For each cell, the pseudo-particle expansion is computed only at the first iteration. After that pseudo-particle positions are extrapolated using the velocities obtained according to eq. (4.7). We set the time step $\Delta t = 0.01$, to be compared with $\min(|\mathbf{v}|/|\mathbf{a}|) \simeq 0.04$.

Fig. 4.9 shows the error percentiles for the configuration as a function of time, compared with the SW data used as a benchmark in the previous sections. Initially the extrapolated pseudo-particles reproduce the real particle dynamical configuration with an accuracy comparable with the static pseudo-particle expansion (the error profile after 0.04 time units is very close to the initial profile). Subsequently, accuracy worsens, and after 0.2 time units errors are considerably larger, with a fraction of large errors increasingly bigger (i.e. an error profile increasingly steeper). From $t = 0$ to $t = 0.2$, the particles having $\epsilon_p < 0.002$ decrease from 99.53% to 64.35%. Particles contributing with large errors to the error profile in fig. 4.9 are those located in the inner core of the distribution. Those are also the particles experiencing higher accelerations. This suggests that the pseudo-particle temporal expansion could be improved by adding a term, similar to eq. (4.7), that accounts for the pseudo-particle acceleration.

4.5 Discussion

In this chapter we presented an error analysis of both the pseudo-particle treecode, and the moving pseudo-particle scheme that we propose to reduce the compute time, and optimise the use of this method with the GRAPE. We showed the error behaviour of this method, and compared it with previous work on the standard treecode.

We modified the pseudo-particle distribution, by adding an extra pseudo-particle located at the cell centre of mass, which led to a one order of magnitude error decrease. Yet, the pseudo-particle scheme tends to be more accurate with homogeneous particle configurations. This is promising for the simulation of systems subject to the Coulomb force, which are usually characterised by nearly uniform density distributions. We also introduced a temporal expansion for the pseudo-particle scheme and showed that, as long as particles are not subject to high accelerations, the error remains close to the error of the standard case up to about 20 time steps. In order to improve the accuracy for particles with high acceleration, we are extending the pseudo-particle temporal expansion to include an acceleration term.

The implementation of a pseudo-particle treecode fine-tuned for the use of the GRAPE is currently under development. The internal architecture of a GRAPE board, where an array of pipelines (up to 96 for GRAPE-4 (Makino *et al.*, 1997), and up to 48 for the most recent GRAPE-6 (Makino *et al.*, 2000), see section 1.3) computes force concurrently on an equal number of particles, is an ideal hardware counterpart of the algorithmic strategies developed to group particles together in order to use the same list of force sources, namely the sinking strategy in Warren & Salmon (1995), or the equivalent grouping strategy in Barnes (1990). Moreover, the internal particle memory of the GRAPE board acts as a cache, which can contain up to about 44 000 particles for the GRAPE-4 (Kawai *et al.*, 1997), and 262 000 particles for the GRAPE-6 (Makino, 2003). Therefore the caching strategies developed in, e.g., Salmon & Warren (1997), where force sources are carefully grouped in logical pages such that data in the same page are likely to be accessed shortly, can be applied here in a natural way. The same force sources set can be used for several reloads of the GRAPE pipelines, and this could considerably reduce the host-GRAPE communication overhead. The use of the pseudo-particle method with the GRAPE can be improved by our pseudo-particle extrapolation scheme. Moreover, this scheme is also suitable for optimal parallelisation, because it allows to retain the multipole expansion of remote cells for a number of iterations, resulting in a substantial decrease of communication among processors.

We aim at using our moving pseudo-particle scheme as the computational core of a parallel treecode running on a hybrid architecture that includes the GRAPE. Performance simulations of this kind of hardware/software configuration were presented in section 3.4.4. We showed there that a pseudo-particle powered treecode running on the GRAPE can outperform the direct code even in case of high accuracy simulations, since the pseudo-particle treecode is able to use the GRAPE also for the evaluation of the force contribution from higher-order multipole terms. In chapter 5, we present a stellar dynamics study of a black hole spiralling in towards the Galactic centre. This study is a first step in the direction of simulating the infall of a star cluster. As discussed in chapter 5 and in section 1.7 above, this problem is very difficult to treat using either the treecode or the direct code. We plan

to develop a hybrid code to solve it. The pseudo-particle treecode is very well suited for playing the role of the treecode "phase" of this hybrid code, especially in view of using the hybrid code on a hybrid architecture including GRAPEs.