# A FAST N-BODY SCHEME FOR COMPUTATIONAL COSMOLOGY 

## by

Marc Victor Leonard Metchnik

A Dissertation Submitted to the Faculty of the DEPARTMENT OF ASTRONOMY<br>In Partial Fulfillment of the Requirements For the Degree of DOCTOR OF PHILOSOPHY In the Graduate College THE UNIVERSITY OF ARIZONA

## The University of Arizona <br> Graduate College

As members of the Dissertation Committee, we certify that we have read the dissertation prepared by Marc Victor Leonard Metchnik entitled A Fast N-body Scheme for Computational Cosmology and recommend that it be accepted as fulfilling the dissertation requirement for the Degree of Doctor of Philosophy.

| Philip A. Pinto | Date: 23 April 2009 |
| :--- | :--- |
|  |  |
| W. David Arnett | Date: 23 April 2009 |

W. David Arnett

Date: 23 April 2009
Romeel A. Davé

Date: 23 April 2009
Daniel J. Eisenstein

Date: 23 April 2009
Peter A. Strittmatter

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copies of the dissertation to the Graduate College.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it be accepted as fulfilling the dissertation requirement.

[^0]
## Statement By Author

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this dissertation are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

Signed: Marc Victor Leonard Metchnik

Table of Contents
List of Figures ..... 5
Abstract ..... 6
Chapter 1 Introduction ..... 7
Chapter 2 Fast Computation of Adaptive Softening Lengths ..... 17
Chapter 3 Periodicity via Direct Summations ..... 29
Chapter 4 Multipole Theory of Infinite Lattice Sums ..... 41
Chapter 5 A Pseudo-Particle Fast Multipole Method ..... 56
Chapter 6 ABACUS: An N-body code for Computational Cosmology ..... 71
Chapter 7 Conclusions and Discussion ..... 90
Appendix A Cartesian-To-Reduced Multipole Transformation ..... 93
Appendix B Cartesian-To-Spherical Harmonic Transformation ..... 95
References ..... 97

## List of Figures

2.1 Adaptive Smoothing Length Algorithm ..... 20
2.2 Coalesced Adaptive Smoothing Length Algorithm ..... 21
2.3 Three Way Sorting ..... 23
2.4 RangeSearching ..... 24
2.5 Pseudocode for Coalesced Interactions ..... 28
4.1 Infinite Lattice ..... 42
4.2 Multipole Expansion Geometry ..... 45
4.3 Distribution of Barnes Hut relative errors ..... 52
4.4 Order 16 single particle Barnes Hut relative errors ..... 52
4.5 Distribution of FMM relative errors ..... 53
4.6 Order 16 single particle FMM relative errors ..... 53
5.1 Monopole Barnes-Hut interaction. ..... 57
5.2 Higher order multipole Barnes-Hut interaction. ..... 58
5.3 Comparison of BH and FMM schemes ..... 59
5.4 Particle-based FMM ..... 60
5.5 Pseudo-particle interaction relative errors ..... 69
6.1 Far Field Geometry ..... 72
6.2 Memory layout for the convolution. ..... 78
6.3 Parallel algorithm for the far field as described in the text. ..... 84
6.4 Far-Field Computational Rate ..... 85
6.5 Cell populations ..... 86
6.6 Zeldovich pancake collapse. ..... 87
6.7 Heitmann $256^{3}$ test problem: our code ..... 88
6.8 Heitmann $256^{3}$ test problem: MC2 ..... 89


#### Abstract

We provide a novel and efficient algorithm for computing accelerations in the periodic large-N-body problem that is at the same time significantly faster and more accurate than previous methods. Our representation of the periodic acceleration is precisely mathematically equivalent to that determined by Ewald summation and is computed directly as an infinite lattice sum using the Newtonian kernel $\left(|\mathbf{r}|^{-1}\right)$. Retaining this kernel implies that one can (i) extend the standard open boundary numerical algorithms and (ii) harness the tremendous computational speed possessed by Graphics Processing Units (GPUs) in computing Newtonian kernels straightforwardly to the periodic domain. The precise form of our direct interactions is based upon the adaptive softening length methodology introduced for open boundary conditions by Price and Monaghan. Furthermore, we describe a new Fast Multipole Method (FMM) that represents the multipoles and Taylor series as collections of pseudoparticles. Using these techniques we have computed forces to machine precision throughout the evolution of a 1 billion particle cosmological simulation with a price/performance ratio more than 100 times that of current numerical techniques operating at much lower accuracy.


## Chapter 1

## Introduction

The gravitational $N$-body problem is to describe the dynamical evolution of a set of $N$ point particles subject to their mutual gravitational interactions. The cosmological N body problem is just the usual gravitational one with two differences. The first is that the canonical problem is set in an expanding background metric given by a solution of the Einstein-De Sitter equations. This is simply a continuous re-scaling of variables and hence does not change the essential character nor mathematics of the solution. The second difference is more fundamental. We require that the solution satisfies, or at least approximates, the cosmological principle of homogeneity and isotropy at the largest scales. In practice, this is achieved by replacing the open boundary conditions of the usual N -body problem with periodic boundary conditions. While this results in, perhaps, a more rigorous enforcement of the principle than one really has in mind, no other mathematical artifice can match the simplicity and precision of its statement, and periodic boundary conditions are universally employed in cosmological N-body simulations. This introduces some fundamental differences.

In the usual gravitational $N$-body problem, which we describe with $N$ particles at positions $\left\{\mathbf{r}_{i}\right\}$ and with masses $\left\{m_{i}\right\}$, the force experienced by a particle, $\mathbf{r}_{i}$, (which we will call the "sink" particle) due to the present of another (a "source"), $\mathbf{r}_{j}$, is given by the familiar direct interaction expression,

$$
\begin{equation*}
\frac{m_{i} m_{j} \mathbf{r}_{i j}}{\left|\mathbf{r}_{i j}\right|^{3}} \tag{1.1}
\end{equation*}
$$

where $\mathbf{r}_{i j}=\mathbf{r}_{i}-\mathbf{r}_{j}$. For the open boundary problem the acceleration at $\mathbf{r}_{i}$ is simply the
sum of the contributions of all of the sources upon each particle in turn,

$$
\begin{equation*}
\mathbf{a}_{i}=-\sum_{\substack{j=1 \\ j \neq i}}^{N} \frac{m_{j} \mathbf{r}_{i j}}{\left|\mathbf{r}_{i j}\right|^{3}} \tag{1.2}
\end{equation*}
$$

Under periodic boundary conditions, the solution becomes more complex; a sink particle "sees" not only each source, but also the infinite number of periodic replicas of each source. One might start by writing down the effect of periodicity as an additional lattice sum over all of the periodic replicas, as a direct summation, i.e.

$$
\begin{equation*}
\mathbf{a}_{i}=-\sum_{\substack{j=1 \\ j \neq i}}^{N} \frac{m_{j} \mathbf{r}_{i j}}{\left|\mathbf{r}_{i j}\right|^{3}}+\sum_{\mathbf{n} \neq(0,0,0)} \sum_{j=1}^{N} \frac{m_{j}\left(\mathbf{r}_{i j}-\mathbf{n}\right)}{\left|\mathbf{r}_{i j}-\mathbf{n}\right|^{3}} \tag{1.3}
\end{equation*}
$$

where $\mathbf{n}$ is a tuple of integers $(a, b, c)$ denoting each individual replicas. Without loss of generality we assume that the simulation volume is unity and centered on the origin. There are two problems with this formulation. First, we have not shown that this is the correct solution to the periodic problem; in fact, while close, it is not quite correct. Second, it is not clear how to perform the infinite sum over replicas in the second term; such sums are well-known to be only conditionally convergent. These problems have driven people to consider different numerical methods for solving the periodic N -body problem; ones which generally employ a representation of the solution in terms of Fourier modes which are a more "natural" representation of periodic functions. I demonstrate in this dissertation that solution methods based upon direct summation are more computationally efficient than other methods currently in use.

A great deal of computational effort has been devoted to studying the formation and evolution of Galactic mass dark matter halos. The largest simulation of the Aquarius project (Springel et al., 2008; Navarro et al., 2008) employed approximately 4.2 billion particles in a $100 h^{-1} \mathrm{Mpc}$ box; it required 3.5 million CPU hours to perform $6.7 \times 10^{13}$
force calculations and used 3TB of memory (800 bytes per particle). It attained an effective computational rate of under 6000 particles/second per CPU. The Millennium-II simulation (Boylan-Kolchin et al., 2009) employed $2160^{3}$ particles within a cubic simulation box of side length $100 h^{-1} \mathrm{Mpc}$. It required 8 TB of memory and 1.4 million CPU hours to evaluate a total of $2.8 \times 10^{13}$ force evaluations for a total of 22,142 simulation time-steps. It attained essentially the same rate and memory usage per simulated particle as the Aquarius simulations. Zemp et al. (2009) report that their Via Lactea II calculation, employing $1.1 \times 10^{9}$ particles in a $40 h^{-1} \mathrm{Mpc}$ box, required 1.1 million CPU hours. The Horizon Run simulation of Kim et al. (2008) simulated 70 billion particles in $\left(6.592 h^{-1}\right.$ Gpe box to model the formation of luminous red galaxies, with the aim of calibrating non-linear gravitational and biasing effects in the future Sloan-III baryon oscillation scale survey; this simulation required 25 days on 412 processors. Similarly, Teyssier et al. (2008) performed a 70 billion particle simulation in a $2 h^{-1} \mathrm{Gpc}$ box with the aim of computing a full-sky weak-lensing convergence map for calibrating future large-scale surveys such as those planned for LSST. This simulation required 3072 processors for 2 months ( 4.4 million CPU hours) and used 18 TB of memory. Clearly, the resources required to perform these computations are very costly. For example, the IBM Roadrunner supercomputer at Los Alamos National Laboratory cost $\$ 133$ million dollars (more than four times the cost of a typical 6.5 m telescope).

In this dissertation we present a scheme for dramatically reducing the computational effort required to perform cosmological N -body simulations.

The state of a cosmological $N$-body simulation is represented by the particles' comoving coordinate vectors $\left\{\mathbf{x}_{i}\right\}$, masses $\left\{m_{i}\right\}$, and corresponding canonical momenta $\mathbf{p}_{i}=a^{2} m_{i} \dot{\mathbf{x}}_{i}$, where $a$ is the cosmological scale factor. Dimensionless variables, $\tilde{x}$ and $\tilde{t}$, may be introduced by scaling the distance by the linear size of the simulation volume and
by measuring the time in units of $H_{0}^{-1}$. The equations of motion to be solved are then

$$
\begin{align*}
\frac{d \tilde{\mathbf{x}}_{i}}{d \tilde{t}} & =a^{-2} \tilde{\mathbf{p}}_{i}  \tag{1.4}\\
\frac{d \tilde{\mathbf{p}}_{i}}{d \tilde{t}} & =a^{-1} \nabla \tilde{\phi}_{i}
\end{align*}
$$

where the evolution of the cosmic scale factor $a$ is given by

$$
\begin{equation*}
\dot{a} \sqrt{a}=H_{0} \sqrt{\Omega_{0}+\Omega_{\mathrm{curv}, 0} a+\Omega_{\Lambda, 0} a^{3}} \tag{1.5}
\end{equation*}
$$

The co-moving acceleration $\nabla \tilde{\phi}_{i}$ is given by a solution of Poisson's equation with periodic boundary conditions. The analytic expression for the co-moving acceleration at position $\mathbf{r}$, first derived by (Ewald, 1921), is, denoting the position of a replica particle $\mathbf{x}_{j}$ in replica $\mathbf{n}$ by $\tilde{\mathbf{x}}_{j, \mathbf{n}}=\tilde{\mathbf{x}}_{j}-\mathbf{n}$,

$$
\begin{align*}
\nabla \tilde{\phi}(\mathbf{r})=\sum_{j=1}^{N} m_{j}\left\{\sum_{\mathbf{n}} \frac{\tilde{\mathbf{x}}_{j, \mathbf{n}}-\mathbf{r}}{\left|\tilde{\mathbf{x}}_{j, \mathbf{n}}-\mathbf{r}\right|^{3}}[ \right. & \operatorname{erfc}\left(\alpha\left|\tilde{\mathbf{x}}_{j, \mathbf{n}}-\mathbf{r}\right|\right) \\
& \left.+\frac{2 \alpha}{\sqrt{\pi}}\left|\tilde{\mathbf{x}}_{j, \mathbf{n}}-\mathbf{r}\right| \mathbf{e}^{-\alpha^{2}\left|\tilde{\mathbf{x}}_{j, \mathbf{n}}-\mathbf{r}\right|^{2}}\right]  \tag{1.6}\\
& \left.+\sum_{\mathbf{k} \neq 0} \frac{2 \mathbf{k}}{|\mathbf{k}|^{2}} \mathbf{e}^{-\pi^{2}|\mathbf{k}|^{2} / \alpha^{2}} \sin \left(2 \pi \mathbf{k} \cdot\left(\tilde{\mathbf{x}}_{j, \mathbf{n}}-\mathbf{r}\right)\right)\right\}
\end{align*}
$$

Contrast this equation to the familiar acceleration at position $\mathbf{r}$ for the open boundary counterpart to our periodic boundary condition solution of Poisson's equation

$$
\begin{equation*}
\nabla \tilde{\phi}(\mathbf{r})=\sum_{j} m_{j} \frac{\mathbf{r}-\tilde{\mathbf{x}}_{j}}{\left|\mathbf{r}-\tilde{\mathbf{x}}_{j}\right|^{3}} \tag{1.7}
\end{equation*}
$$

This is a vastly simpler than expression (1.6) and leads us to ask, What is the total number of operations required to obtain the force for a particle to machine precision (i.e a relative error of $10^{-6}$, irrespective of the distribution the surrounding particles).

While the final resolution to this question is beyond the scope of this dissertation, we have made significant progress towards its solution. The question is really concerned with
the optimal computational representation of the periodic solution of Poisson's equation. Before we consider different representations of periodicity, we remind the reader of the intriguing difference between the mathematical formulation of periodicity and it's open counterpart; despite the fact that the periodic solution also admits a purely Newtonian $\left(|\mathbf{r}|^{-1}\right)$ direct summation representation the asymptotic scaling of the periodic solution is $O\left(N^{3 / 2}\right)$ in contrast to the familiar $O\left(N^{2}\right)$ scaling of it's non-periodic counterpart. An insight into the nature of this scaling was particularly well illustrated by Fincham, D. (1994), and we present a precis of his derivation below.

Consider a cubic simulation box of side $L$ containing $N$ particles, and assume that $L$ varies as $N^{1 / 3}$. Recall that the real space part of the Ewald potential involves the function $\operatorname{erfc}\left(\alpha r_{i j}\right) / r_{i j}$. Asymptotically $\operatorname{erfc}(\alpha r)$ behaves as $\exp \left(-\alpha^{2} r^{2}\right)$. Assuming that we wish to bound the real space term by $\varepsilon=\exp (-p)$, we thus require $\alpha^{2} R^{2} \approx p$, where $R$ is the real space cutoff distance corresponding to a tolerance of $\varepsilon$. The time required to evaluate the real space sum, for all $N$ particles, is $T_{R}=\frac{1}{2} N \frac{4 \pi}{3} R^{3} n t_{R}$, where $n$ is the number density.

The reciprocal space involves a term dominated by the term $\exp \left(-k^{2} / 4 \alpha^{2}\right)$; if we again require this term to be bounded by $\varepsilon$ at the reciprocal space cutoff $K$, then we require $p=K^{2} /\left(4 \alpha^{2}\right)$. The time required to evaluate the reciprocal space points is given by $\frac{4 \pi}{3} K^{3} \frac{L^{3}}{\left(8 \pi^{3}\right)}$ using the fact that the volume per reciprocal space point is $(2 \pi / L)^{3}$ since $\mathbf{k}=\frac{2 \pi}{L}(l, n, n)$. Writing $L^{3}=N / n$, then and using the expression for $K$ in terms of $p$ above, $T_{F}=\frac{1}{2} \frac{4 \pi}{3}(p / \pi)^{3} N^{2} /\left(n R^{3}\right) t_{F}$. The total execution time is

$$
\begin{equation*}
T=\frac{1}{2} \frac{4 \pi}{3}\left[N n R^{3} t_{R}+\left(\frac{p}{\pi}\right)^{3} \frac{N^{2}}{n R^{3}} t_{F}\right] \tag{1.8}
\end{equation*}
$$

Setting $d T / d R=0$, the value of $R$ which minimizes the above expression is

$$
\begin{equation*}
R_{O P T}=\left(\frac{p}{\pi}\right)^{\frac{1}{2}}\left(\frac{t_{F}}{t_{R}}\right)^{\frac{1}{6}} \frac{N^{\frac{1}{6}}}{n^{\frac{1}{3}}} \tag{1.9}
\end{equation*}
$$

Using this $R$ in the equation above for $T$ we get

$$
\begin{equation*}
T_{O P T}=\frac{4 \pi}{3} N^{3 / 2} \frac{p}{\pi}^{3 / 2}\left(t_{R} t_{F}\right)^{1 / 2} \tag{1.10}
\end{equation*}
$$

This $O\left(N^{3 / 2}\right)$ behavior exhibits a property of periodicity which is akin to Debye shielding; i.e the net result is a reduction the effective interaction between particles to a short-range "screened" Coulomb interaction. Naturally one is lead to the tantalizing notion that each particle need only interact with only $O(\sqrt{N})$ particles, e.g. in a $512^{3}$ particle simulation, each particle would only need to interact with $\approx 12,000$ other particles in order to compute an acceleration to machine precision!

More generally, the previous derivation indicates that fast-summation algorithms can be applied to periodic problems in addition to the open boundary problems to which they have been applied. Fast-summation techniques (Barnes and Hut, 1986; Greengard and Rokhlin, 1987) rely upon the artifice of partitioning each particle's interactions into near and far field sets. Since the force on a particle due to nearby particles will fluctuate strongly with position, a smooth function will not be able accurately to represent their aggregate effect. Thus the contribution from nearby particles must thus be determined by summing the exact expression for the acceleration they induce (with or without a "softening" of the force law at small distances). Smooth functions can, however, represent the force due to a collection of particles at sufficiently large distance. For the far field, one can use so-called multipole expansions of varying orders, for all particles which are sufficiently well separated from the test particle( (where the concept of order of an expansion corresponds to the highest degree of an approximation polynomial in the Taylor series expansion of a function). This notion of "well-separatedness" is the crucial desideratum for applying the multipole expansion. Geometrically (and somewhat informally), two sets of particles are well-separated if each set can be circumscribed within a sphere, if the
two spheres do not intersect, and if the distance between the centers of these spheres is larger than the sum of their radii by a factor of order unity. The computational efficiency of fast-summation methods comes from the fact that the contributions from subsets of the far field sources can be aggregated into single interactions. Thus, the interactions between a large number of distant particles can be represented by a smaller number of aggregate interactions. Provided the work done in aggregating is smaller than performing the equivalent number of direct interactions, a computational speed-up can be obtained.

Whilst fast summation is a critical component of our program, the question remains as to what mathematical expression one partitions into near and far fields. The application of fast summation to the expression for the Ewald potential was implemented by Stadel (2001). The problem with their approach is that the expression for the Ewald acceleration which one must evaluate for all near field interactions is very expensive. In contrast, if we were to represent the periodic acceleration as a direct interaction scheme, this would allow the periodic acceleration to be computed using only the Newtonian $1 / r^{2}$ gravitational force between two point masses (which we will call a "direct interaction"), as if the problem had open, not periodic, boundary conditions. To do this, we represent the periodic acceleration at position $\mathbf{r}$ as as

$$
\begin{gather*}
\Psi(\mathbf{r})=\sum_{j=1}^{N} \frac{m_{j}\left(\mathbf{r}_{j}-\mathbf{r}\right)}{\left|\mathbf{r}_{j}-\mathbf{r}\right|^{3}}+\sum_{\mathbf{n}=(0,0,0)} \sum_{j=1}^{N} \frac{m_{j}\left(\mathbf{r}_{j, \mathbf{n}}-\mathbf{r}\right)}{\left|\mathbf{r}_{j, \mathbf{n}}-\mathbf{r}\right|^{3}} \\
-\frac{4 \pi}{3}\left[\sum_{j} m_{j} \mathbf{r}_{j}-\mathbf{r} \sum_{j} m_{j}\right] \tag{1.11}
\end{gather*}
$$

In this expression the first term is the familiar open boundary problem; the remaining terms represent the contribution from an infinite periodic lattice of replicas of the volume being simulated, where as before $\mathbf{r}_{j, \mathbf{n}}=\mathbf{r}_{j}-\mathbf{n}$, with $\mathbf{n}$ a lattice vector denoting the particular replica in the sum. We will demonstrate in Chapter 2 that this acceleration is
mathematically equivalent to the Ewald acceleration introduced earlier. Comparing the Ewald representation and this direct sum over replicas formulation, one can easily see that the latter might well be much faster to compute, provided that the infinite sum over replicas can be handled efficiently. To do this, we exploit the translational symmetry implicit in periodic boundary conditions and the considerable simplification afforded by recognizing that the mass distribution, by definition, is identical in each periodic replica and can therefore be removed from the lattice sum analytically.

Most current cosmological simulation codes (GOTPM, Dubinski et al. 2004; PMFAST, Merz et al. 2005; GRACOS, Shirokov and Bertschinger 2005; GADGET2, Springel 2005; MC2, Heitmann et al. 2005; to name but a few) employ Fourier methods to represent periodic solutions of Poisson's equation. The most venerable technique in this category is the Particle Mesh (PM) scheme. The density field represented by the particles is first interpolated onto a mesh, and the potential is then computed using the fast Fourier transform (FFT). Schematically,

$$
\nabla^{2} \Phi(\mathbf{r})=\rho(\mathbf{r})=\sum_{\mathbf{k}} \hat{\rho}(\mathbf{k}) e^{i \mathbf{k} \cdot \mathbf{r}} \rightarrow \mathbf{k}^{2} \hat{\Phi}(\mathbf{k})=\hat{\rho}(\mathbf{k}) \rightarrow \Phi(\mathbf{r})=\sum_{\mathbf{k}} \frac{\hat{\rho}(\mathbf{k})}{\mathbf{k}^{2}} e^{i \mathbf{k} \cdot \mathbf{r}}
$$

where the carat above a quantity refers to its Fourier transform. The solution represented in this manner is a straightforward convolution of the density field and a Green's function $\left(1 / \mathbf{k}^{2}\right)$. The periodic acceleration is then just the gradient of this potential, interpolated at the positions of the particles. This method suffers from the limited resolution afforded by representing the density on a regular grid and whilst the scheme has been extended to adaptive meshes (such as the adaptive P3M scheme (AP3M) ) it is unable to obtain periodic accelerations to machine precision.

A further advance came from considering the Ewald potential which is the analytic solution to Poisson's equation with periodic boundary conditions whose kernel is written
as

$$
\begin{equation*}
G_{\text {Ewald }}(\mathbf{r})=\sum_{\mathbf{n}} \frac{\operatorname{erfc}(\alpha|\mathbf{x}-\mathbf{n}|)}{|\mathbf{x}-\mathbf{n}|}-\sum_{\mathbf{h} \neq 0} \frac{1}{\pi|\mathbf{h}|^{2}} \exp \left(-\frac{\pi^{2}|\mathbf{h}|^{2}}{\alpha^{2}}\right) \cos (2 \pi \mathbf{h} \cdot \mathbf{x}) \tag{1.12}
\end{equation*}
$$

Rybicki (1986) recognized that the $\mathbf{h}$-space term in the Ewald potential might most easily be calculated in Fourier space while the real-space term (the sum over n) could be evaluated directly. He thus split the Green's function into two terms

$$
\begin{align*}
\Phi_{k} & =-\frac{4 \pi G \rho_{k}}{k^{2}} \\
& =-\frac{4 \pi G \rho_{k}}{k^{2}} \exp \left(-k^{2} r_{s}^{2}\right)-\frac{4 \pi G \rho_{k}}{k^{2}}\left(1-\exp \left(-k^{2} r_{s}^{2}\right)\right)  \tag{1.13}\\
& =\Phi_{k}^{l}+\Phi_{k}^{s}
\end{align*}
$$

where $\Phi_{k}^{l}$ and $\Phi_{k}^{s}$ are the long and short range potential, respectively. The long range potential is computed in Fourier space, just as in a PM code, but using the kernel $\exp \left(-k^{2} r_{s}^{2}\right) / k^{2}$ instead of the standard $1 / k^{2}$ kernel, whereas the short-range force is computed using a kernel of the form

$$
\begin{equation*}
\xi(\mathbf{r})=-\frac{G m \mathbf{r}}{r^{3}}\left(\operatorname{erfc}\left(\frac{r}{2 r_{s}}\right)+\frac{r}{r_{s} \sqrt{\pi}} \exp \left(-\frac{r^{2}}{4 r_{s}^{2}}\right)\right) \tag{1.14}
\end{equation*}
$$

over volumes of a few Fourier grid cells. Both TreePM (Bagla, 2002) and Gadget-II and -III (Springel, 2005) codes implement this scheme. We will show that the direct summation scheme is preferable to any of these Fourier schemes.

The only aspect which remains is to quantify our original claim that we can dramatically reduce the computational effort required to perform N -body simulations. We will demonstrate in this dissertation that a direct summation representation of periodicity along with commodity Graphics Processing Units (GPU's) (which can compute 45 billion direct interactions per second) can reduce the cost of large computations by at least
a factor of 100 ; typically a factor of at least 10 from the direct summation representation and a factor of 10 from the GPU. This is not merely a hypothetical conjecture. Consider the results of Merz (2006) for his CubePM code on the BlueGene/L supercomputer for a simulation which employed $1024^{3}$ particles in a box 1 Gpc on a side. CubePM uses a two-level particle-mesh scheme and is designed to run on massively parallel computers, using a distributed Fast Fourier Transform (the 3DFFT Library of M. Eleftheriou et al. 2005). Merz reports that the wallclock time per time step (averaged over the first 10 steps) was 20 seconds on 4096 BlueGene/L processors. Given that the 65,000 processors on this machine cost $\$ 100$ million, the 4096 processors used effectively cost $\$ 6.3$ million. In contrast, on a desktop computer which currently costs $\$ 6,000$, we have performed the same calculation in 50 seconds, and to very much higher accuracy. Even ignoring this increased accuracy, we have demonstrated an increase in price/performance by a factor of more than 400 , turning a supercomputer problem into a routine desktop calculation. This incredible price/performance increase is the result of using new algorithms tailored to commodity hardware, and especially to our effective use of GPUs. We have already used this code to produced 70 simulations with one billion particles in a box $1 \mathrm{~h}^{-1} \mathrm{Gpc}$ on a side which have been used to study non-linear shifts of the baryon acoustic oscillation scale and to provide a large set of mock catalogs for our analysis of the SDSS and BOSS data sets.

## CHAPTER 2

## Fast Computation of Adaptive Softening Lengths

If one is to make a faithful simulation of the evolution of an N -body system, it is essential to make an accurate computation of the periodic acceleration $\Omega(\mathbf{r})$. Equally important, however, is representing the evolution of a collision-less fluid with a (relatively) small number of particles. The smaller the number of particles, the more strong (large-angle) scattering introduces a spurious collisionality to the simulation. One remedy is simply to increase the number of particles in the simulation; the greater the number of particles, the less often any particle's near field is dominated by a single other particle. More practically however, one must "soften" the interaction at small inter-particle separations, thereby lessening the effect of large-angle scattering.

Force softening is crucial since ultimately one is concerned not with computing N body trajectories of individual particles, but rather with simulating the solution of the coupled Vlasov and Poisson equations for a continuous, collisionless fluid. Two-body encounters artificially introduce collisional relaxation, as otherwise, for any finite $N$, there would always be a closest particle which would dominate a given particle's acceleration. Force softening is a technique used to minimize this effect, by modifying the force so that as the inter-particle separation falls below some specified length, the softening length, the force is bounded by a constant value. All of the N -body simulations mentioned previously use a constant co-moving softening length. Because bound structures cannot form within the softening length, in the quest to resolve ever-smaller structures a softening length is often chosen appropriate to the highest densities the simulation will ever produce. The result is that this length is too small for most of the volume in the simulation, which then experiences excessive two-body relaxation. This in turn affects the density profiles of
halos and the halo mass function.
The choice of optimal softening length has been discussed at length (Athanassoula et al., 2000; Dehnen, 2001; Price and Monaghan, 2007). These studies were carried out in the context of isolated hales in dynamical equilibrium using analytic distributions so that it is possible to compute the relative error in various physical quantities with analytical expressions derived from the distribution functions. Dehnen (2001) concluded that the optimal softening length must adapt to the local inter-particle separation as a function of space and time. We incorporate the formulation of variable softening length proposed by Price and Monaghan (2007) which conserves momentum and energy explicitly. The modified force kernel is now given by

$$
\begin{equation*}
\phi^{\prime}=\frac{4 \pi}{r^{2}} \int_{0}^{r} W r^{\prime 2} d r^{\prime} \tag{2.1}
\end{equation*}
$$

and $W(r, h)$ has compact support, e.g. the cubic spline kernel of Monaghan \& Lattanzio (1985)

$$
W(r, h)= \begin{cases}\frac{1}{\pi h^{3}}\left[1-\frac{3}{2}\left(\frac{r}{h}\right)^{2}+\frac{3}{4}\left(\frac{r}{h}\right)^{3}\right] & \frac{r}{h}<1  \tag{2.2}\\ \frac{1}{\pi h^{3}}\left[\frac{1}{4}\left(2-\frac{r}{h}\right)^{3}\right] & 1 \leq \frac{r}{h}<2 \\ 0 & \frac{r}{h} \geq 2\end{cases}
$$

may be written in the form

$$
\phi^{\prime}(r, h)= \begin{cases}\frac{1}{h^{2}}\left(\frac{4}{3}\left(\frac{r}{h}\right)-\frac{6}{5}\left(\frac{r}{h}\right)^{3}+\frac{1}{2}\left(\frac{r}{h}\right)^{4}\right) & \frac{r}{h}<1  \tag{2.3}\\ \frac{1}{h^{2}}\left(\frac{8}{3}\left(\frac{r}{h}\right)-3\left(\frac{r}{h}\right)^{2}+\frac{6}{5}\left(\frac{r}{h}\right)^{3}-\frac{1}{6}\left(\frac{r}{h}\right)^{4}-\frac{1}{15}\left(\frac{h}{r}\right)^{2}\right) & 1 \leq \frac{r}{h}<2 \\ \frac{1}{r^{2}} & \frac{r}{h} \geq 2\end{cases}
$$

where $h$ is the softening length.

Mathematically, the softened acceleration is given by the expression,

$$
\begin{align*}
\mathbf{a}_{i}= & G \sum_{j} m_{j}\left[\frac{\phi_{i j}^{\prime}\left(h_{i}\right)+\phi_{i j}^{\prime}\left(h_{j}\right)}{2}\right] \frac{\mathbf{r}_{i}-\mathbf{r}_{j}}{\left|\mathbf{r}_{i}-\mathbf{r}_{j}\right|}  \tag{2.4}\\
& -\frac{G}{2}\left[\frac{\zeta_{i}}{\Omega_{i}} \frac{\partial W_{i j}\left(h_{i}\right)}{\partial \mathbf{r}_{i}}+\frac{\zeta_{j}}{\Omega_{j}} \frac{\partial W_{i j}\left(h_{j}\right)}{\partial r_{j}}\right]
\end{align*}
$$

where

$$
\begin{equation*}
\zeta_{i}=\frac{\partial h_{i}}{\partial \rho_{i}} \sum_{j} m_{j} \frac{\partial \phi_{i j}\left(h_{i}\right)}{\partial h_{i}}, \quad \Omega_{i}=1-\frac{\partial h_{i}}{\partial \rho_{i}} \sum_{j} m_{j} \frac{\partial W_{i j}\left(h_{i}\right)}{\partial h_{i}} \tag{2.5}
\end{equation*}
$$

and where

$$
\frac{d W(r, h)}{d h}= \begin{cases}\frac{1}{\pi h^{4}}\left[-3+\frac{15}{2}\left(\frac{r}{h}\right)^{2}-\frac{9}{2}\left(\frac{r}{h}\right)^{3}\right] & \frac{r}{h}<1  \tag{2.6}\\ \frac{1}{\pi h^{4}} \frac{3}{2}\left(2-\frac{r}{h}\right)^{2}\left(\frac{r}{h}-1\right) & 1 \leq \frac{r}{h}<2 \\ 0 & \frac{r}{h} \geq 2\end{cases}
$$

and

$$
\frac{d W(r, h)}{d \mathbf{r}}= \begin{cases}\frac{3}{\pi h^{4}} \frac{\mathbf{r}}{r}\left(-\left(\frac{r}{h}\right)+\frac{3}{4}\left(\frac{r}{h}\right)^{2}\right) & \frac{r}{h}<1  \tag{2.7}\\ \frac{3}{\pi h^{4}} \frac{\mathbf{r}}{r}\left(\frac{-1}{4}\left(2-\left(\frac{r}{h}\right)\right)^{2}\right) & 1 \leq \frac{r}{h}<2 \\ 0 & \frac{r}{h} \geq 2\end{cases}
$$

and the derivative of the potential with respect to $h$ is given by

$$
\frac{\partial \phi}{\partial h}= \begin{cases}\frac{1}{h^{2}}\left(-2\left(\frac{r}{h}\right)^{2}+\frac{3}{2}\left(\frac{r}{h}\right)^{4}-\frac{3}{5}\left(\frac{r}{h}\right)^{5}+\frac{7}{5}\right) & \frac{r}{h}<1  \tag{2.8}\\ \frac{1}{h^{2}}\left(-4\left(\frac{r}{h}\right)^{2}+4\left(\frac{r}{h}\right)^{3}-\frac{3}{2}\left(\frac{r}{h}\right)^{4}+\frac{1}{5}\left(\frac{r}{h}\right)^{5}+\frac{8}{5}\right) & 1 \leq \frac{r}{h}<2 \\ 0 & \frac{r}{h} \geq 2\end{cases}
$$

Note that the sum over $j$ in the expression for the acceleration $\mathbf{a}_{i}$ only contains nonzero terms for a small subset of particles due to the fact that $h_{i}, h_{j}$ is typically chosen to contain around 60 particles. The algorithm (Price and Monaghan, 2007) employed for setting the softening length is outlined in pseudocode in Figure (2.1). The gist of the algorithm to obtain $h$ may be summarized as using a Newton-Raphson iteration scheme

$$
\begin{aligned}
& \text { Algorithm AdaptiveSofteningLength }\left(\mathbf{r}, \varepsilon, h_{0}, N_{\text {neighbours }}, \sigma\right) \\
& \qquad \begin{aligned}
& f=f^{\prime}=1 \\
& h=h_{0} \\
& \text { while }\left(f>f^{\prime} \varepsilon\right) \\
&\{b\}=\left\{i \cdot\left|\mathbf{r}-\mathbf{r}_{i}\right|<2 h\right\} \\
& \rho_{\Sigma}(\mathbf{r}, h)=\sum_{\{b\}} W\left(\left|\mathbf{r}-\mathbf{r}_{b}\right|, h\right) \\
& \Omega_{\Sigma}(\mathbf{r}, h)=\sum_{\{b\}} \frac{\partial W\left(\left|\mathbf{r}-\mathbf{r}_{b}\right|, h\right)}{\partial h} \\
& \rho=\left(\frac{\eta}{h}\right)^{3} \\
& \Omega=1+\frac{h}{3 \rho} \Omega_{\Sigma}(\mathbf{r}, h) \\
& f=\rho-\rho_{\Sigma}(\mathbf{r}, h) \\
& f^{\prime}=3 \frac{\rho}{h} \Omega \\
& h=h-\frac{f}{f^{\prime}} \\
& \text { endwhile }
\end{aligned}
\end{aligned}
$$

Figure 2.1: Adaptive Smoothing Length Algorithm
on the functional

$$
\begin{equation*}
\left(\frac{\eta}{h}\right)^{3}-\rho_{\Sigma}(h) \tag{2.9}
\end{equation*}
$$

where the first term reflects a rough relation between the smoothing length and the density and $\rho_{\Sigma}(h)$ is the current value of the density computed via the usual SPH smoothing estimate using the current value of $h . \eta$ defined as parameter which specifies the smoothing length in terms of average inter-particle spacing. $\eta$ is related to the desired number of neighbours via $N_{\text {neighbours }}=\frac{4 \pi}{3}(\sigma \eta)^{3}$ and $\sigma$ is the compact support radius of the kernel ( $=2$ for the cubic spline), $\varepsilon$ is a convergence tolerance for the Newton-Raphson

Algorithm CoalescedAdaptiveSofteningLength $\left(S_{n_{i}}, \varepsilon, h_{0_{i}}, N_{\text {neighbours }}, \sigma\right)$

$$
\begin{aligned}
& S^{\prime}=S_{n_{i}} \\
& f=f^{\prime}=1 \\
& \forall j \in S^{\prime} \quad h_{j}=h_{0, j} \\
& \text { while }\left(\left|S^{\prime}\right|>0\right) \\
& \vec{c}=\frac{\sum_{j \in S^{\prime}} \mathbf{r}_{j}}{\left|S^{\prime}\right|} \\
& h=\max _{j \in S^{\prime}} h_{j} \\
& r=\max _{j \in S^{\prime}}\left|\mathbf{r}_{j}-\vec{c}\right| \\
& \{b\}=\text { RangeSearch }(\vec{c}, r+2 h) \\
& \forall j \in S^{\prime} \rho_{\Sigma, j}=\sum_{\{b\}} W\left(\left|\mathbf{r}_{j}-\mathbf{r}_{b}\right|, h_{j}\right) \\
& \forall j \in S^{\prime} \Omega_{\Sigma, j}=\sum_{\{b\}} \frac{\partial W\left(\left|\mathbf{r}_{j}-\mathbf{r}_{b}\right|, h_{j}\right)}{\partial h} \\
& \forall j \in S^{\prime} \quad\{ \\
& \rho_{j}=\left(\frac{\eta}{h_{j}}\right)^{3} \\
& \Omega_{j}=1+\frac{h i_{j}}{3 \rho_{j}} \Omega_{\Sigma, j} \\
& f_{j}=\rho_{j}-\rho_{\Sigma, j} \\
& f_{j}^{\prime}=3 \frac{\rho_{j}}{h_{j}} \Omega_{j} \\
& h_{j}=h_{j}-\frac{f_{j}}{f_{j}^{\prime}} \\
& \text { if }\left(f_{j}<f_{j}^{\prime} \varepsilon\right) S^{\prime}=S^{\prime}-\{j\} \\
& \text { \} } \\
& \text { endwhile }
\end{aligned}
$$

Figure 2.2: Coalesced Adaptive Smoothing Length Algorithm
iteration and $h_{0}$ is an initial guess for the softening length, say the distance to the nearest neighbour. A naive implementation to obtain $\rho_{\Sigma}$ and $\Omega_{\Sigma}$ for all $N$ particles would led to an $O\left(N^{2}\right)$ algorithm, if the quantity $\{b\}$, which is the set of particles within a radius of $2 h$ of the particle at location $\mathbf{r}$, could not be obtained in less than $O(N)$ time.

Clearly, to start, we require an algorithm to obtain all particles within radius $2 h$ of the current particle under consideration, in at worst $O(\log N)$ time. This is usually referred to as a range search. In two and three dimensions the $k d$-tree introduced by Jon Bentley in 1975, is the de-facto standard data structure employed for range searching. Trees are a hierarchical representation of a point set, i.e the set is recursively subdivided into a subsets (nodes) $V_{i}$ such that $V=\bigcup V_{i}$ and $V_{i} \cap V_{j}=0$ with each subset $V_{i}$ also satisfying this property. Nodes are notionally volumes if the set comprises three dimensional particle coordinates. The kd-tree is a binary tree (each node is subdivided into two nodes) where each subset (node) is recursively bisected via an axis aligned splitting hyperplane. The canonical method of cycling the axis of the splitting plane $(x, y, z, x, y, z, \cdots, x, y, z)$ at a location such that an equal number of particles lie in each sub-volume leads to a balanced kd-tree.

Unfortunately spatial data structures which involve quasi-random memory accesses are not particularly suited to the GPU due to the penalty of main memory accesses. Instead, we employ a non-hierarchical data structure based purely upon triply sorting the particle set. Our algorithm is based upon the canonical one dimensional scheme for finding all particles within radius $R$ from a given position $x$, namely to sort the particles based on their co-ordinate, and then binary search for the indices within the sorted list closes to $x-R$ and $x+R$. In two dimensions, assuming we have $N^{2}$ particles, sort the particles on the $x$-coordinates and then bin the particles into $N$ bins. Within each of the $N$ bins sort on the $y$-coordinate. Consider a particle $\phi=\left(p_{x}, p_{y}\right)$; find the $x$ bin which contains the $x$

```
Algorithm 3waySort(N, particles)
sort( \&(particles[0]), \&(particles[np]), ParticleSortX() );
xbegin \([0]=-0.5\);
\(\boldsymbol{f o r}(\boldsymbol{i n t} \mathrm{i}=0 ; \mathrm{i}<\mathrm{ns} ; \mathrm{i}++\) ) \{
    \(\mathbf{i f}(\mathrm{i}<\mathrm{ns}-1)\{\)
        xend \([\mathrm{i}]=\) particles[ns \(\times \mathrm{ns} \times(\mathrm{i}+1)] . \mathrm{x}\);
        xbegin \([i+1]=\) xend[i];
    \}
    else
        xend[ns-1] \(=\frac{1}{2}\);
    sort(\& \((\) particles[ns \(\times n s \times i]), \&(\) particles[ns \(\times n s \times(i+1)])\), ParticleSortY ()\()\);
    \(\operatorname{ybegin}[\mathrm{i} \times \mathrm{ns}]=-\frac{1}{2}\)
    for(int \(\boldsymbol{j}=0 ; \mathrm{j}<\mathrm{ns} ; \mathrm{j}++\) ) \(\{\)
        if( \(\mathrm{j}<\mathrm{ns}-1\) ) \(\{\)
            yend \([\mathrm{i} \times \mathrm{ns}+\mathrm{j}]=\operatorname{particles}[\mathrm{ns} \times \mathrm{ns} \times \mathrm{i}+\mathrm{ns} \times(\mathrm{j}+1)] . \mathrm{y}\);
            ybegin \([\mathrm{i} \times \mathrm{ns}+(\mathrm{j}+1)]=\) yend \([\mathrm{i} \times \mathrm{ns}+\mathrm{j}]\);
        \}
        else
            \(\operatorname{yend}[\mathrm{i} \times \mathrm{ns}+(\mathrm{ns}-1)]=\frac{1}{2} ;\)
        sort(\& (particles[i \(\times \mathrm{ns} \times \mathrm{ns}+\mathrm{j} \times \mathrm{ns}]), \&(\) particles \([i \times n s \times n s+(j+1) \times n s])\),
            ParticleSortZ() );
    \}
\}
```


## Figure 2.3: Pseudocode for a 3 way sort

coordinate value of $p_{x}-R$ denotes this bin $X^{-}$; and obtain the bin $X^{+}$(corresponding to the bin containing $p_{x}+R$ ). For each bin $X$ from $X^{-}$to $X^{+}$, form the two indices $Y_{X}^{-}$and $Y_{X}^{+}$, which are within $\pm R$ of $p_{y}$. Any particle within the indices $Y_{X}^{-} \cdots Y_{X}^{+}$in bin $X$ can be tested for inclusion in the circle for radius $R$ surrounding $\phi$. The generalization of both the sorting scheme and range searching to three dimensions is completely straightforward, and shown in Figures (2.3) and (2.4).

## Algorithm RangeSearch(r, R, S)

$$
\begin{aligned}
& X^{-}=\text {BinarySearch } X\left(r_{x}-\mathrm{R}\right) \\
& X^{+}=\text {BinarySearchX }\left(r_{x}+\mathrm{R}\right) \\
& \text { for }\left(\mathrm{x}=X^{-} ; \mathrm{x} \leq X^{+} ; \mathrm{x}++\right) \text { \{ } \\
& Y^{-}=\operatorname{BinarySearchY}\left(r_{y}-\mathrm{R}, \mathrm{x}\right) \\
& Y^{+}=\operatorname{BinarySearchY}\left(r_{y}+\mathrm{R}, \mathrm{x}\right) \\
& \text { for }\left(\mathrm{y}=Y^{-} ; \mathrm{y}<=Y^{+} ; \mathrm{y}++\right)\{ \\
& Z^{-}=\operatorname{BinarySearchZ}\left(r_{z}-\mathrm{R}, \mathrm{x}, \mathrm{y}\right) \text {; } \\
& Z^{+}=\text {BinarySearchZ }\left(r_{z}+\mathrm{R}, \mathrm{x}, \mathrm{y}\right) \text {; } \\
& \text { for }\left(p=Z^{-} ; p<=Z^{+} ; p++\right)\{ \\
& \text { if(Distance } \left.\left(\mathbf{r}, \mathbf{r}_{p}\right) \leq \mathrm{R}\right) \mathrm{S}=\mathrm{S} \cup\{\mathrm{p}\} ;
\end{aligned}
$$

Figure 2.4: Pseudocode for RangeSearch
Whilst our algorithm reduces the $O\left(N^{2}\right)$ naive implementation of $\rho_{\Sigma}$ and $\Omega_{\Sigma}$ to $O(N \log N)$ unfortunately it exhibits low arithmetic intensity - a high memory access ratio to computation ratio (typical of spatial data structure traversals). In other words, arithmetic intensity is the ratio of the cycles devoted to floating point operations to cycles required to transfer a memory word; consequently, large register sets and caches are necessary to obviate the penalty associated with memory accesses as cached variables enhance arithmetic intensity through the rescue of variables without the penalty of a memory access. High arithmetic intensity can be exposed thorough a generic pattern (template) we refer to as coalesced interaction, whereby for two sets, $\left\{x_{i}\right\},\left\{y_{j}\right\}$, the quantity $\sum_{j} K\left(x_{i}, y_{j}\right)$ is accumulated into $\left\{a_{i}\right\}$, as outlined in Figure (2.5) using the NVIDIA CUDA extensions to the C programming language.

The computational rate for coalesced interactions; which include computing and accumulating direct accelerations or nearest neighbours, or in the present case softening
lengths can be obtained quite precisely as $F /(k i j)$ interactions per second, where kernel $K(x, y)$ requires $k$ flops and the total number of flops available is denoted by $F$. For the NVIDIA GTX 280 GPU $F \approx 10^{12}$ flops and interactions for obtaining distances between particles (squared distances are actually employed to avoid the sqrt operation) require $k=$ 11 cycles. Clearly we need to reformulate the computation of $\rho_{\Sigma}$ and $\Omega_{\Sigma}$ as a sequence of purely local operations based upon sets of points, rather than individual particles. Querying our range searching data structure in this manner for a coalesced set of particles obviates the overheads associated with traversing the data structure for each point individually. Consider the set of $N$ particles $S=\left\{\mathbf{r}_{1}, \cdots, \mathbf{r}_{N}\right\}$, and select a subset $S_{n}=\left\{\mathbf{r}_{i_{1}}, \cdots, \mathbf{r}_{i_{n}}\right\}$. Define dual $\left(S_{n}\right)=\left\{\mathbf{r}_{i_{1}}^{\prime}, \cdots, \mathbf{r}_{i_{m}}^{\prime}\right\}, \forall \mathbf{r} \in S \wedge \mathbf{r}_{i} \in S_{n} \cdot\left|\mathbf{r}-\mathbf{r}_{i}\right|<2 h_{i} \rightarrow \mathbf{r} \in \operatorname{dual}\left(S_{n}\right)$ where we intend the notation to imply that $\left|S_{n}\right|=n$ and $\left|\operatorname{dual}\left(S_{n}\right)\right|=m$. Naturally $n \leq m \leq N$, however if $S_{n}$ was randomly selected would expect that even for $n \ll N, m \approx N$; clearly for $n \ll N$ one desires $m \ll N$. We desire a partitioning of $S$, into $k$ subsets, i.e

$$
\begin{equation*}
\bigcup_{i=1}^{k} S_{n_{i}}=S, \quad \sum_{i=1}^{k} n_{i}=N, \quad S_{i} \cap S_{j}=\phi \quad 1 \leq i<j \leq k \tag{2.10}
\end{equation*}
$$

and we seek to minimize

$$
\begin{equation*}
W_{n_{i}}=\sum_{i}\left|S_{n_{i}}\right|\left|\operatorname{dual}\left(S_{n_{i}}\right)\right| \tag{2.11}
\end{equation*}
$$

$W_{n_{i}}$ is a metric which quantifies the amount of work to compute $\rho_{\Sigma}\left(S_{n_{i}}\right)$ and $\Omega_{\Sigma}\left(S_{n_{i}}\right)$. Given a set $S_{n_{i}}$ we form

$$
\begin{align*}
& \vec{c}=\frac{\sum_{j} m_{j} \mathbf{r}_{j}}{\sum_{j} m_{j}}, \mathbf{r}_{j} \in S_{n_{i}} \\
& r=\max _{j}\left|\mathbf{r}_{j}-\vec{c}\right|, \quad \mathbf{r}_{j} \in S_{n_{i}}  \tag{2.12}\\
& h=\max _{j} h_{j}, \quad \mathbf{r}_{j} \in S_{n_{i}}
\end{align*}
$$

and then use a range search query to obtain all particles centered on $\vec{c}$ within a radius or $r+2 h$ an approximation to dual $\left(S_{n_{i}}\right)$. In order to select subsets which are spatially compact spherical regions we employ the Mutually Nearest Neighbour (MNN) tree structure introduced by Press 1986. Informally, the MNN tree is constructed by determining all particles nearest neighbours and then replacing mutually nearest neighbours with another particle at the center of mass of those particles. The process is repeated until only one particle remains. In this manner this tree has $2 N-1$ nodes, where $N$ is the number of particles. With an MNN tree one could select $S_{n_{i}}$ by a simple recursive scheme whereby the algorithm descends the MNN tree, $T$, and if the amount of work $W_{T}$ is greater than the pre-specified amount $m$, then the daughters $W_{T_{R}}$ and $W_{T_{L}}$ are recursively traversed. The construction of the MNN tree itself is completely amenable to our coalesced interactions; sets, $S$, are obtained from the octree representation of the $N$ particles and the set dual $(S)$ is similarly formed where $h$ is the maximum of the nearest neighbour distances in $S$. The octree is a tree data structure in which each internal node has up to eight children, and moreover, each node in an octree subdivides the space it represents into eight equal sub-octants, irrespective of the particle distribution within the volume. The octree data structure is based upon the idea of mapping a three dimensional co-ordinate into a one dimension key using a space filling curve, and subsequently range partitioning this one dimensional key hierarchically across a set of nodes. We construct the octree by first forming the Morton order (introduced by G.M. Morton in 1966) of each particle by interleaving the binary representations of integer representations of the floating point coordinate values of the particles. Once the particles are sorted into this ordering the resulting ordering is equivalent to a depth-first traversal of an octree.

As an illustrative example, the effective rate of computation of these softening lengths for a Plummer distribution was almost 5 million particles per second on a GTX 295 for
a maximum set size of 800 particles. We also mention that the technique of coalesced interactions is well suited to both the SPH smoothing length computation (essentially identical to our algorithm to obtain the softening lengths) and for the evaluation of SPH dynamical quantities. As a first step toward the goal of creating a cosmological SPHNbody code for the GPU, we implemented a non-cosmology "classic" SPH-Nbody code and performed the Evrard collapse test ( a standard test for SPH codes including selfgravity - the test follows the adiabatic collapse of an initial cold and static gas sphere with an ideal gas equation of state). Disregarding the time to compute the smoothing lengths we could compute the SPH fluid quantities for the gas particles at just over 4 million particles per second on a GTX 295.

Clearly, as only the nearest interactions involve smoothing, it will not be necessary in the remainder of this dissertation, devoted to the computation of the far field interactions, to include the effects of smoothing.

```
Algorithm Coalescedlnteraction( float4 *g_xi, float4 *g_xj,
int ni, int nj, float \(\left.4 * g \_a c c u m u l a t o r\right) ~\{\)
\begin{tabular}{rlll} 
const & unsigned & int tid & \(=\) threadldx.x; \\
const & unsigned & int bid & \(=\) blockldx.x; \\
const & & int nblocks & \(=\) gridDim.x; \\
const & & int nthreadsperblock & \(=\) blockDim.x; \\
& unsigned & int \(n\) _jblock & \(=\) nj/CACHESIZE; \\
& unsigned & int \(n\) _iblock & \(=\) ni/(nblocks \(\times\) nthreadsperblock);
\end{tabular}
for(unsigned int iloop \(=0\); iloop \(<n\) _iblock; iloop ++ ) \{
    unsigned int gindex \(=(\) iloop \(\times\) nblocks + bid \() \times\) nthreadsperblock + tid;
    g_accumulator[gindex] \(=0\);
\}
__syncthreads();
__shared__ float4 shared_xj[CACHESIZE];
for (unsigned int jloop \(=0\); jloop < n_jblock; jloop++) \{
    unsigned int j_start \(=\) CACHESIZE timesjloop;
    __syncthreads();
    for(unsigned int \(j=0 ; j<\) CACHESIZE; \(j+=n t h r e a d s p e r b l o c k) ~\{\)
        unsigned int \(\mathrm{jj}=\mathrm{j}+\) tid;
        shared_xj \([j \mathrm{j}]=\) g_xi \([\mathrm{j}\) _start +jj\(]\);
    \}
    __syncthreads();
        for (unsigned int iloop \(=0\); iloop <n_iblock; iloop++) \{
            unsigned int \(\mathrm{i}=(\) iloop \(\times\) nblocks + bid \() \times\) nthreadsperblock + tid;
            float xi ;
            \(x i=g \_x i[i] ;\)
            aprevious = g_accumulator[i];
            a. \(\mathrm{x}=\mathrm{a} . \mathrm{y}=\mathrm{a} . \mathrm{z}=\mathrm{a} . \mathrm{w}=0\);
            for(unsigned int \(\mathbf{j}=0 ; \mathrm{j}<\) CACHESIZE; \(\mathrm{j}++\) )
            Accumulate(xi, shared_xj[j], \&a);
            g_accumulator \([i]=\mathrm{a}+\) aprevious
        \}
    \}
\}
```

Figure 2.5: Pseudocode for Coalesced interactions

## Chapter 3

## Periodicity via Direct Summations

Historically, when solving the gravitational N -body problem, the change from isolated to periodic boundary conditions has engendered a completely different effective local interaction potential (Gadget-2, PkdGrav). Why should it be that simply changing the distant boundary conditions should require one to modify the local interaction potential? This question is especially important given that the $1 / r$ kernel can be computed so efficiently. We are not the first to desire a representation based purely upon direct summations. Bouchet and Hernquist (1988) proposed a scheme which they dubbed QuasiPeriodic (QP) boundaries; this was actually a variation of the minimum-image convention of Alder and Wainwright (1959) used in molecular dynamics. The ansatz behind the QP method was that the acceleration on a mass can be computed as if it were embedded in an infinite periodic system. Instead of computing the infinite sum over replicas, however, the minimum-image convention includes only the contribution of those masses which lie within a volume the size of the simulation volume, centered about each mass. Bouchet and Hernquist (1988) concluded that this method was not sufficiently exact; for example, it induced too-rapid growth of the fundamental mode in cosmological simulations. They ascribed this behavior to the fact that the method did not take into account the force contribution from more distant replicas. In a neutral plasma Debye screening suppresses longer range interactions so the minimum-image convention is intuitively sensible. For gravity, with its long range forces, it is not surprising that a direct summation representation which only considered the nearest neighbouring replicas should underestimate the total acceleration.

Hernquist et al. (1991) subsequently turned to tabulating the Ewald correction to the open-boundary potential, already common practice in computational chemistry (e.g. Sangster and Dixon, 1976). This table-lookup was later adopted by Dubinski (1996), Davé et al. (1997), and Springel et al. (2001) (in Gadget-1). One of the distinct advantages of a tabulated Ewald correction was that it permitted a computational implementation of periodicity using direct interactions. Direct interactions enabled the incorporation of GRAPE (GRAvity piPE) boards which were special-purpose ASIC (application specific integrated circuit) dedicated to the computation of a Newtonian $1 / r^{2}$ gravitation force between two point masses (Sugimoto et al., 1990; Fukushige et al., 2005). The dominance the GRAPE enjoyed during the last decade has essentially been superseded by the superior price-performance afforded by commodity CPU's and Graphics Processing Units (GPU's) for direct interactions. Our implementation of a direct interaction a quad-core Intel Q9950 processor can compute 2.4 billion direct accelerations with Plummer smoothing per second, an effective rate of 4.4 clock cycles per acceleration. On the NVIDIA GTX 295 GPU, we compute the same quantity at a sustained rate of 45 billion direct interactions per second, at the same cost per part as the CPU. The new Intel i7 CPU's and the next generation of GPU's both deliver more than double these rates.

Whilst all modern cosmological simulations assume periodic boundary conditions, this does not mean we expect that the universe is indeed periodic. Rather, when simulating a "typical" volume of the universe, the effect of the rest of the universe on the computed volume must be included. The simulation volume is the only sample of structure available; a natural way to include its effects is by replicating it infinitely in all directions. This notion of infinite replication is the physical picture behind what what is colloquially referred to by "periodic boundary conditions," regardless of the mathematical language in which the conditions are expressed.

One might write the acceleration due to all masses and their periodic images as the infinite sum ${ }^{1}$

$$
\begin{equation*}
\Psi(\mathbf{r})=\sum_{j=1}^{N} \frac{m_{j}\left(\mathbf{r}_{j}-\mathbf{r}\right)}{\left|\mathbf{r}_{j}-\mathbf{r}\right|^{3}}+\sum_{|\mathbf{n}|^{2}>0}^{\infty} \sum_{j=1}^{N} \frac{m_{j}\left(\mathbf{r}_{j}-\mathbf{n}-\mathbf{r}\right)}{\left|\mathbf{r}_{j}-\mathbf{n}-\mathbf{r}\right|^{3}} \tag{3.1}
\end{equation*}
$$

Here we have explicitly expressed the acceleration as a sum of the contribution from the unit cell (the volume being simulated) and from its infinite replicas. What are the consequences of imposing periodic boundary conditions upon a finite set of masses by replicating the simulation volume ad infinitum in this way? Is the problem even mathematically well-posed?

As noted in the previous chapter, most modern cosmological simulations have resorted to Fourier methods to represent periodicity. Proving the expected equivalence of Fourier methods and infinite replication has been the source of much controversy for decades. While one's intuition suggests that infinite replication should be equivalent to the construction of periodicity through Fourier methods, the precise formulation of this equivalence is subtle. In this chapter we shall show how periodic boundary conditions may efficiently be included in N -body computations using direct summation.

Ewald (1921) developed an alternate view of the infinite sum in (3.1) in which the density distribution is not represented on a lattice. The derivation of Ewald's representation proceeds directly from the Fourier space Green's function,

$$
\begin{equation*}
G\left(\mathbf{r}, \mathbf{r}^{\prime}\right)=\sum_{\mathbf{k} \neq \mathbf{0}} \frac{1}{|2 \pi \mathbf{k}|^{2}} \mathbf{e}^{2 \pi i \mathbf{k} \cdot\left(\mathbf{r}-\mathbf{r}^{\prime}\right)} \tag{3.2}
\end{equation*}
$$

and the relation

$$
\begin{equation*}
\frac{1}{|\mathbf{k}|^{2}}=\int_{0}^{\infty} \mathbf{e}^{-\left(\mathbf{k}^{2}\right) t} d t \tag{3.3}
\end{equation*}
$$

[^1]Using this expression, the Fourier Green's function can be written as

$$
\begin{equation*}
G\left(\mathbf{r}, \mathbf{r}^{\prime}\right)=\sum_{\mathbf{k} \neq \mathbf{0}} \frac{\mathrm{e}^{2 \pi \mathrm{i} \mathbf{k} \cdot\left(\mathbf{r}-\mathbf{r}^{\prime}\right)}}{|2 \pi \mathbf{k}|^{2}}=\sum_{\mathbf{k} \neq \mathbf{0}} \int_{0}^{\infty} \mathrm{e}^{-(2 \pi \mathbf{k})^{2} t+2 \pi i \mathbf{k} \cdot\left(\mathbf{r}-\mathbf{r}^{\prime}\right)} \mathrm{d} t \tag{3.4}
\end{equation*}
$$

Recalling the Poisson-Jacobi formula,

$$
\begin{equation*}
\sum_{k=-\infty}^{\infty} \mathbf{e}^{-(2 \pi k)^{2} t+2 \pi i k\left(x-x^{\prime}\right)}=\frac{1}{\sqrt{4 \pi t}} \sum_{K=-\infty}^{\infty} \mathbf{e}^{-\frac{\left(x-x^{\prime}+K\right)^{2}}{4 t}} \tag{3.5}
\end{equation*}
$$

we have an alternative expression for the Green's function,

$$
\begin{equation*}
G\left(\mathbf{r}, \mathbf{r}^{\prime}\right)=\sum_{\mathbf{K}} \int_{0}^{\infty}(5 \pi t)^{-\frac{3}{2}} \mathbf{e}^{-\frac{\left(\mathbf{r}-\mathbf{r}^{\prime}+K\right)^{2}}{4 t}} \mathrm{~d} t \tag{3.6}
\end{equation*}
$$

These two representations of $G\left(\mathbf{r}, \mathbf{r}^{\prime}\right)$ are useful when $t$ is large and when $t$ is small respectively. The familiar Ewald form can be recovered by splitting the integral over $t$ in $G\left(\mathbf{r}, \mathbf{r}^{\prime}\right)$ into two domains which utilize these representations,

$$
\begin{align*}
G\left(\mathbf{r}, \mathbf{r}^{\prime}\right)= & \int_{0}^{\alpha^{2}} \sum_{\mathbf{K}}(4 \pi t)^{-\frac{-3}{2}} \mathbf{e}^{-\frac{\left.\mathbf{r}-\mathbf{r}^{\prime}+K\right)^{2}}{4 t}} \mathrm{~d} t \\
& +\int_{\alpha^{2}}^{\infty}\left[\sum_{\mathbf{k} \neq 0} \mathrm{e}^{-(2 \pi \mathbf{k})^{2} t+2 \pi i \mathbf{k} \cdot\left(\mathbf{r}-\mathbf{r}^{\prime}\right)}\right] \mathrm{d} t  \tag{3.7}\\
= & \frac{1}{4 \pi} \sum_{\mathbf{k}} \frac{\operatorname{erfc}\left(\left|\mathbf{r}-\mathbf{r}^{\prime}+\mathbf{k}\right| / \sqrt{2} \alpha\right)}{\left|\mathbf{r}-\mathbf{r}^{\prime}+\mathbf{k}\right|}-\alpha^{2} \\
& +\sum_{\mathbf{k} \neq \mathbf{0}} \frac{\mathbf{e}^{-\alpha^{2}(2 \pi \mathbf{k})^{2}+2 \pi i \mathbf{k} \cdot\left(\mathbf{r}-\mathbf{r}^{\prime}\right)}}{(2 \pi \mathbf{k})^{2}}
\end{align*}
$$

For $N$ particles with positions $\mathbf{r}_{j}$ and masses $m_{j}$ in a unit volume centered about the origin,
the "Ewald acceleration" which is derived from $G\left(\mathbf{r}, \mathbf{r}^{\prime}\right)$ is usually written in the form

$$
\left.\begin{array}{rl}
\Omega(\mathbf{r})=\sum_{j=1}^{N} m_{j}\{ & \sum_{\mathbf{n}} \frac{\mathbf{r}_{j, \mathbf{n}}-\mathbf{r}}{\left|\mathbf{r}_{j, \mathbf{n}}-\mathbf{r}\right|^{3}}[
\end{array} \operatorname{erfc}\left(\alpha\left|\mathbf{r}_{j, \mathbf{n}}-\mathbf{r}\right|\right), ~\left(\frac{2 \alpha}{\sqrt{\pi}}\left|\mathbf{r}_{j, \mathbf{n}}-\mathbf{r}\right| \mathbf{e}^{-\alpha^{2}\left|\mathbf{r}_{j, \mathbf{n}}-\mathbf{r}\right|^{2}}\right]\right\}
$$

where $\mathbf{r}_{j, \mathbf{n}}=\mathbf{r}_{j}+\mathbf{n}$. The Ewald acceleration is thus in some sense the discrete particle analog of the Fourier acceleration defined for a smooth distribution. Having derived the Ewald acceleration from the Fourier Green's function we have shown the equivalence between the Fourier and Ewald representations of the solution to Poisson's equation under periodic boundary conditions, we will now demonstrate the relationship between the Ewald representation and one based upon infinite direct summation as in (3.1).

The principle difficulty which arises with such sums on an infinite lattice is that they are, in general, not absolutely convergent; the asymptotic limit of these sums is dependent upon the order of summation. The study of the acceleration due to an infinite assembly of point particles has roots going back at least to Gibbs and Wilson (1901), though its mathematical nuance has only been completely settled within the last quarter-century. de Leeuw et al. (1980) proved that the naïve sum over replicas (3.1) is convergent if summed over spherical shells surrounding the simulation volume. Certainly this nuance was appreciated by Peebles (1980) who stated for infinite distributions: "the sum in general is not well-defined; the answer depends on how the terms are ordered. The prescription ... is that the sum is in order of increasing $\left|\mathbf{x}_{j}-\mathbf{x}\right|$. Under the assumption that the particle distribution is a spatially homogeneous and isotropic random process ... this sum converges to a definite value well within the relativistic horizon." Whilst summation over spherical shells can indeed converge to a definite value, this is not in general the same value given
by Fourier methods; an additional term is required for the acceleration (3.1) to coincide with the standard Fourier result.

The emphasis above on can is seen in the nuance of the computation of Madelung's constant for the NaCl crystal, a simple and wide-studied lattice sum. Following (Borwein et al., 1985), the computation of Madelung's constant requires an evaluation of the sum of the elements in the following set

$$
\begin{equation*}
B=\left\{\frac{(-1)^{i+j+k}}{\sqrt{i^{2}+j^{2}+k^{2}}}:(i, j, k) \in Z^{3} /(0,0,0)\right\} \tag{3.9}
\end{equation*}
$$

Let $C(n)$ denote the number of ways of writing $n$ as a sum of three squares. If we consider a sphere centered at the origin in three-space, add all elements that correspond to lattice points within the sphere, and then let the radius go to infinity, one is led to the divergent sum

$$
\begin{equation*}
\sum_{n=1}^{\infty} \frac{(-1)^{n} C(n)}{\sqrt{n}} \tag{3.10}
\end{equation*}
$$

Remarkably, however, if we sum over expanding cubes,

$$
\begin{equation*}
S(n)=\sum_{(i, j, k) \neq(0,0,0)} \frac{(-1)^{i+j+k}}{\left(i^{2}+j^{2}+k^{2}\right)^{1 / 2}} \quad-n \leq i, j, j \leq n \tag{3.11}
\end{equation*}
$$

the limit $\lim _{n \rightarrow \infty} S(n)$ exists, (Borwein et al., 1985).
In a particularly insightful paper, Redlack and Grindlay (1975) (hereafter RG75) demonstrated the relation between the acceleration from an infinitely-replicated simulation volume $\Psi\left(\mathbf{r}_{i}\right)$ and its representation via Ewald summation, $\Omega(\mathbf{r})$. Scaling the sim-
ulation to unit volume and centering it about the origin, their result is

$$
\begin{align*}
\Omega(\mathbf{r})= & \sum_{j=1}^{N} \frac{m_{j}\left(\mathbf{r}_{j}-\mathbf{r}\right)}{\left|\mathbf{r}_{j}-\mathbf{r}\right|^{3}}+\sum_{|\mathbf{n}| \neq 0} \sum_{j=1}^{N} \frac{m_{j}\left(\mathbf{r}_{j, \mathbf{n}}-\mathbf{r}\right)}{\left|\mathbf{r}_{j, \mathbf{n}}-\mathbf{r}\right|^{3}} \\
& -\frac{4 \pi}{3}\left[\sum_{j} m_{j} \mathbf{r}_{j}-\mathbf{r} \sum_{j} m_{j}\right]  \tag{3.12}\\
= & \Psi(\mathbf{r})-\zeta_{R G}(\mathbf{r})
\end{align*}
$$

The last term, $\zeta_{R G}(\mathbf{r})$, is the additional term alluded to above which we shall call the Redlack-Grindlay term. Hence, the naïve summation over all replicas is identical to the Ewald acceleration when a term related to the dipole moment of the unit cell is included, provided that the lattice sum in $\Psi(\mathbf{r})$ is computed in increasing spherical shells. Since an appreciation of the origin of the $\zeta_{R G}$ term in the infinite replication of a simulation volume of a finite number of particles is so pertinent to our presentation we provide here a précis of the proofs in de Leeuw et al. (1980) and in Smith (1981).

Recalling the equality

$$
\begin{equation*}
\frac{1}{|\mathbf{r}|}=\frac{1}{\sqrt{\pi}} \int_{0}^{\infty} t^{-\frac{1}{2}} \mathbf{e}^{-t \mathbf{r}^{2}} \mathrm{~d} t \tag{3.13}
\end{equation*}
$$

one can rewrite the sum of the potential over replicas as

$$
\sum_{\mathbf{n} \in \mathfrak{F}_{R}} \frac{1}{|\mathbf{r}+\mathbf{n}|}=\sum_{\mathbf{n} \in \mathfrak{F}_{R}} \frac{1}{\sqrt{\pi}} \int_{0}^{\infty} t^{-\frac{1}{2}} \mathbf{e}^{-t(\mathbf{r}+\mathbf{n})^{2}} \mathrm{~d} t
$$

where $\mathfrak{P}_{R}$ denotes the set of tuples $\{(a, b, c):-R \leq a, b, c \leq R\}$. Now split the range of integration as in the derivation of the Ewald representation

$$
\begin{equation*}
\sum_{\mathbf{n} \in \mathfrak{P}_{R}} \frac{1}{|\mathbf{r}+\mathbf{n}|}=\sum_{\mathbf{n} \in \mathfrak{P}_{R}} \frac{1}{\sqrt{\pi}}\left[\int_{0}^{\alpha^{2}} t^{-\frac{1}{2}} \mathbf{e}^{-t(\mathbf{r}+\mathbf{n})^{2}} \mathrm{~d} t+\int_{\alpha^{2}}^{\infty} t^{-\frac{1}{2}} \mathbf{e}^{-t(\mathbf{r}+\mathbf{n})^{2}} \mathrm{~d} t\right] \tag{3.14}
\end{equation*}
$$

Using the identity

$$
\begin{equation*}
\mathbf{e}^{-t a^{2}}=(\pi t)^{\frac{1}{2}} \int_{-\infty}^{\infty} \mathbf{e}^{-u^{2} / t+2 i u a} \mathrm{~d} u \tag{3.15}
\end{equation*}
$$

we have

$$
\begin{equation*}
\mathbf{e}^{-t(\mathbf{r}+\mathbf{n})^{2}}=(\pi t)^{-\frac{3}{2}} \int_{V_{\infty}} \mathbf{e}^{-\mathbf{u}^{2} / t+2 i \mathbf{r} \cdot \mathbf{u}} \mathbf{e}^{2 i \mathbf{n} \cdot \mathbf{u}} \mathrm{~d} \mathbf{u} \tag{3.16}
\end{equation*}
$$

where $V_{\infty}$ denotes all of three dimensional space. The first term can be simplified as

$$
\begin{align*}
\frac{1}{\sqrt{\pi}} \int_{0}^{\alpha^{2}} t^{-\frac{1}{2}} & \left(\sum_{\mathbf{n} \in \mathfrak{P}_{R}} \mathbf{e}^{-t(\mathbf{r}+\mathbf{n})^{2}}\right) \mathrm{d} t \\
& =\frac{1}{\sqrt{\pi}} \int_{0}^{\alpha^{2}} t^{-\frac{1}{2}}\left(\sum_{\mathbf{n} \in \mathfrak{P}_{R}}(\pi t)^{-\frac{3}{2}} \int_{V_{\infty}} \mathbf{e}^{-\mathbf{u}^{2} / t+2 i \mathbf{r} \cdot \mathbf{u}} \mathbf{e}^{2 i \mathbf{n} \cdot \mathbf{u}} \mathrm{~d} \mathbf{u}\right) \mathrm{d} t  \tag{3.17}\\
& =\frac{1}{\pi^{2}} \int_{V_{\infty}} \mathbf{e}^{2 i \mathbf{r} \cdot \mathbf{u}}\left(\int_{0}^{\alpha^{2}} t^{-2} \mathbf{e}^{-\mathbf{u}^{2} / t} \mathrm{~d} t\right) \sum_{\mathbf{n} \in \mathfrak{P}_{R}} \mathbf{e}^{2 i \mathbf{n} \cdot \mathbf{u}} \mathrm{~d} \mathbf{u} \\
& =\frac{1}{\pi^{2}} \int_{V_{\infty}} \frac{1}{\mathbf{u}^{2}} \mathbf{e}^{-\mathbf{u}^{2} / \alpha^{2}+2 i \mathbf{r} \cdot \mathbf{u}} \sum_{\mathbf{n} \in \mathfrak{P}_{R}} \mathbf{e}^{2 i \mathbf{n} \cdot \mathbf{u}} \mathrm{~d} \mathbf{u}
\end{align*}
$$

The second term in (3.14) can be simplified by recalling the definition of the error function and the identity

$$
\begin{equation*}
\frac{1}{\pi} \int_{\alpha^{2}}^{\infty} t^{-\frac{1}{2}} \mathbf{e}^{-t \mathbf{r}^{2}} \mathrm{~d} t=\frac{\operatorname{erfc}(\alpha|\mathbf{r}|)}{|\mathbf{r}|} \tag{3.18}
\end{equation*}
$$

so that the lattice sum of the potential becomes

$$
\begin{align*}
\sum_{\mathbf{n} \in \mathfrak{P}_{R}} \frac{1}{|\mathbf{r}+\mathbf{n}|}= & {\left[\frac{1}{\pi^{2}} \int_{V_{\infty}} \frac{1}{\mathbf{u}^{2}} \mathbf{e}^{-\mathbf{u}^{2} / \alpha^{2}+2 i \mathbf{r} \cdot \mathbf{u}} \sum_{\mathbf{n} \in \mathfrak{P}_{R}} \mathbf{e}^{2 i \mathbf{n} \cdot \mathbf{u}} \mathrm{du}\right.} \\
& +\left[f(\mathbf{r}, \alpha)+\sum_{\substack{\mathbf{n} \in \mathfrak{P}_{R} \\
\mathbf{n} \neq \mathbf{0}}} \frac{\operatorname{erfc}(\alpha|\mathbf{r}+\mathbf{n}|)}{|\mathbf{r}+\mathbf{n}|}\right] \tag{3.19}
\end{align*}
$$

where

$$
f(\mathbf{r}, \alpha)= \begin{cases}\operatorname{erfc}(\alpha|\mathbf{r}|) /|\mathbf{r}| & \mathbf{r} \neq \mathbf{0}  \tag{3.20}\\ -2 \alpha / \sqrt{\pi} & \mathbf{r}=\mathbf{0}\end{cases}
$$

There is a clear resemblance between this expression and the kernel of Ewald potential,

$$
\begin{equation*}
G_{\text {Ewald }}(\mathbf{x})=\sum_{\mathbf{n}} \frac{\operatorname{erfc}(\alpha|\mathbf{x}-\mathbf{n}|)}{|\mathbf{x}-\mathbf{n}|}-\sum_{\mathbf{h} \neq 0} \frac{1}{\pi|\mathbf{h}|^{2}} \exp \left(-\frac{\pi^{2}|\mathbf{h}|^{2}}{\alpha^{2}}\right) \cos (2 \pi \mathbf{h} \cdot \mathbf{x}) \tag{3.21}
\end{equation*}
$$

Consider

$$
\begin{equation*}
\frac{1}{\pi^{2}} \int_{V_{\infty}} \frac{1}{\mathbf{u}^{2}} \mathbf{e}^{-\mathbf{u}^{2} / \alpha^{2}+2 i \mathbf{r} \cdot \mathbf{u}} \sum_{\mathbf{n} \in \mathfrak{P}_{N}} \mathbf{e}^{2 i \mathbf{n} \cdot \mathbf{u}} \mathrm{~d} \mathbf{u} \tag{3.22}
\end{equation*}
$$

where $\mathbf{u}=\pi \mathbf{m}$ where $\mathbf{m}$ is on $\mathfrak{R}_{R}$, the reciprocal lattice to $\mathfrak{B}_{R}$. The reciprocal lattice vector $\mathbf{m}=\left(m_{1}, m_{2}, m_{3}\right)$ is related to a lattice vector $\mathbf{n}=\left(n_{1}, n_{2}, n_{3}\right)$ as $m_{i}=2 \pi / n_{i}$. Upon substituting $\mathbf{v}+\pi m=\mathbf{u}$ into (3.22) we have

$$
\begin{align*}
& \frac{1}{\pi^{2}} \int_{V_{\infty}} \frac{1}{\mathbf{u}^{2}} \mathbf{e}^{-\mathbf{u}^{2} / \alpha^{2}+2 i \mathbf{r} \cdot \mathbf{u}} \sum_{\mathbf{n} \in \mathfrak{P}_{N}} \mathbf{e}^{2 i \mathbf{n} \cdot \mathbf{u}} \mathrm{~d} \mathbf{u} \\
&=I_{\mathbf{0}}\left(\mathbf{r}, \mathfrak{P}_{R}\right)+\sum_{\substack{\mathbf{m} \in \mathfrak{R}_{R} \\
\mathbf{m} \neq \mathbf{0}}} \mathbf{e}^{-\pi^{2} \mathbf{m}^{2} / \alpha^{2}+2 \pi i \mathbf{m} \cdot \mathbf{r}} I_{\mathbf{m}}\left(\mathbf{r}, \mathfrak{P}_{R}\right) \tag{3.23}
\end{align*}
$$

where $\Omega_{0}=[-\pi / 2, \pi / 2]^{3}$, and we have defined

$$
\begin{equation*}
I_{\mathbf{m}}\left(\mathbf{r}, \mathfrak{P}_{R}\right)=\frac{1}{\pi^{2}} \int_{\Omega_{0}}\left(\frac{e^{-\mathbf{v}^{2} / \alpha^{2}-2 \pi i \mathbf{m} \cdot \mathbf{v} / \alpha^{2}+2 i \mathbf{r} \cdot \mathbf{v}}}{|\pi \mathbf{m}+\mathbf{v}|^{2}} Q(\mathbf{v})\right) \mathrm{d} \mathbf{v} \tag{3.24}
\end{equation*}
$$

and

$$
\begin{equation*}
Q(\mathbf{v})=\sum_{\mathbf{n} \in \mathfrak{P}_{R}} e^{2 i \mathbf{n} \cdot \mathbf{v}}=\prod_{j=1}^{3} \sum_{n=-N}^{N} e^{2 i n v_{j}}=\prod_{j=1}^{3} \frac{\sin \left((2 N+1) v_{j}\right)}{\sin v_{j}} \tag{3.25}
\end{equation*}
$$

Expression (3.24) may then be evaluated by introducing the function $g_{\mathbf{m}}(\mathbf{v}, \mathbf{r})$ for $\mathbf{m} \neq \mathbf{0}$,

$$
\begin{equation*}
g_{\mathbf{m}}(\mathbf{v}, \mathbf{r})=|\pi m+\mathbf{v}|^{-2} e^{-\mathbf{v}^{2} / \alpha^{2}-2 \pi i \mathbf{m} \cdot \mathbf{v} / \alpha^{2}+2 i \mathbf{r} \cdot \mathbf{v}} \tag{3.26}
\end{equation*}
$$

so that,

$$
\begin{align*}
& I_{\mathbf{m}}\left(\mathbf{r}, \mathfrak{P}_{N}\right) \\
& =\frac{1}{\pi^{2}} \int_{\Omega_{0}} g_{\mathbf{m}}(\mathbf{0}, \mathbf{r}) \sum_{\mathbf{n} \in \mathfrak{P}_{N}} \mathbf{e}^{2 i \mathbf{n} \cdot \mathbf{v}} \mathrm{~d} \mathbf{v}+\frac{1}{\pi^{2}} \int_{\Omega_{0}}\left(g_{\mathbf{m}}(\mathbf{v}, \mathbf{r})-g_{\mathbf{m}}(\mathbf{0}, \mathbf{r})\right) \sum_{\mathbf{n} \in \mathfrak{P}_{N}} \mathbf{e}^{2 i \mathbf{n} \cdot \mathbf{v}} \mathrm{~d} \mathbf{v} \\
& =\frac{1}{\pi^{4} \mathbf{m}^{2}} \prod_{j=1}^{3} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \frac{\sin \left((2 N+1) v_{j}\right)}{\sin v_{j}}  \tag{3.27}\\
& +\pi \sum_{\mathbf{n} \in \mathfrak{P}_{N}} \frac{1}{\pi} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} d v_{1} \mathbf{e}^{2 i n_{1} v_{1}} \frac{1}{\pi} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} d v_{2} \mathbf{e}^{2 i n_{2} v_{2}} \frac{1}{\pi} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} d v_{3} \mathbf{e}^{2 i i_{3} v_{3}}\left(g_{\mathbf{m}}(\mathbf{v}, \mathbf{r})-g_{\mathbf{m}}(\mathbf{0}, \mathbf{r})\right)
\end{align*}
$$

The first term in the expression above is $\left(\pi \mathbf{m}^{2}\right)^{-1}$. The second term is the partial sum of the three-dimensional Fourier series for $g_{\mathbf{m}}(\mathbf{v}, \mathbf{r})-g_{\mathbf{m}}(\mathbf{0}, \mathbf{r})$, evaluated at $\mathbf{v}=\mathbf{0}$. Since the function is analytic at and around this point, the partial sum is equal to the function evaluated at this point plus terms that are $O\left((2 N+1)^{-2}\right)$. Hence,

$$
\begin{equation*}
I_{\mathbf{m}}\left(\mathbf{r}, \mathfrak{P}_{N}\right)=\left(\pi \mathbf{m}^{2}\right)^{-1}+O\left((2 N+1)^{-2}\right) \tag{3.28}
\end{equation*}
$$

Finally, the Redlack-Grindlay term, $\zeta_{R G}$ arises from the derivative of

$$
\begin{equation*}
I_{\mathbf{0}}\left(\mathbf{r}, \mathfrak{P}_{N}\right)=I_{\mathbf{0}}\left(\mathbf{0}, \mathfrak{P}_{N}\right)+\frac{1}{\pi^{2}} \int_{T_{0}} d^{3} \mathbf{v} \frac{e^{-\mathbf{v}^{2} / \beta^{2}}\left(e^{2 i \mathbf{r} \cdot \mathbf{v}}-1\right)}{\mathbf{v}^{2}} Q(\mathbf{v}) \tag{3.29}
\end{equation*}
$$

Rewriting the second factor in the integrand as

$$
\begin{equation*}
e^{2 i \mathbf{r} \cdot \mathbf{v}}-1=-2 \sin ^{2}(\mathbf{r} \cdot \mathbf{v})+i \sin (2 \mathbf{r} \cdot \mathbf{v}) \tag{3.30}
\end{equation*}
$$

we see that, since the term $i \sin (2 \mathbf{r} \cdot \mathbf{v})$ is odd in $\mathbf{v}$, the integral of this term over $d^{3} \mathbf{v}$ vanishes. Thus,

$$
\begin{align*}
\int_{T_{0}} d^{3} \mathbf{v} \frac{e^{-\mathbf{v}^{2} / \beta^{2}}\left(e^{2 i \mathbf{r} \cdot \mathbf{v}}-1\right)}{\mathbf{v}^{2}} Q(\mathbf{v})= & -\int_{T_{0}} d^{3} \mathbf{v} \frac{\sin ^{2}(\mathbf{r} \cdot \mathbf{v})}{\mathbf{v}^{2}} Q(\mathbf{v}) \\
& +\int_{T_{0}} d^{3} \mathbf{v} \sin ^{2}(\mathbf{r} \cdot \mathbf{v}) \frac{e^{-\mathbf{v}^{2} / \beta^{2}}-1}{\mathbf{v}^{2}} Q(\mathbf{v}) \tag{3.31}
\end{align*}
$$

The second term in the sum above tends to $O\left((2 N+1)^{-2}\right)$ for precisely the same reasons as the integral in (3.27). In addition to this truncation order, we can approximate $\sin ^{2}(\mathbf{r} \cdot \mathbf{v})$ with $(\mathbf{r} \cdot \mathbf{v})^{2}$, the first term of its Taylor series around $\mathbf{v}=\mathbf{0}$. We thus obtain,

$$
\begin{equation*}
I_{\mathbf{0}}\left(\mathbf{r}, \mathfrak{P}_{N}\right)=I_{\mathbf{0}}\left(\mathbf{0}, \mathfrak{P}_{N}\right)-\frac{2}{\pi^{2}} \int_{T_{0}} d^{3} \mathbf{v} \frac{(\mathbf{r} \cdot \mathbf{v})^{2}}{\mathbf{v}^{2}} Q(\mathbf{v})+O\left((2 N+1)^{-2}\right) \tag{3.32}
\end{equation*}
$$

If we expand $(\mathbf{r} \cdot \mathbf{v})^{2}$,

$$
\begin{equation*}
(\mathbf{r} \cdot \mathbf{v})^{2}=\left(r_{1} v_{1}\right)^{2}+2 r_{1} v_{1} r_{2} v_{2}+\left(r_{2} v_{2}\right)^{2}+2 r_{1} v_{1} r_{3} v_{3}+2 r_{2} v_{2} r_{3} v_{3}+\left(r_{3} v_{3}\right)^{2} \tag{3.33}
\end{equation*}
$$

we see that all odd terms (i.e. cross terms), when integrated over all $\mathbf{v}$, will sum to zero, and therefore the integral expression above evaluates to

$$
\begin{equation*}
\frac{2}{\pi^{2}}\left[r_{1}^{2} \int_{T_{0}} d^{3} \mathbf{v} \frac{v_{1}^{2}}{\mathbf{v}^{2}} Q(\mathbf{v})+r_{2}^{2} \int_{T_{0}} d^{3} \mathbf{v} \frac{v_{2}^{2}}{\mathbf{v}^{2}} Q(\mathbf{v})+r_{3}^{2} \int_{T_{0}} d^{3} \mathbf{v} \frac{v_{3}^{2}}{\mathbf{v}^{2}} Q(\mathbf{v})\right] \tag{3.34}
\end{equation*}
$$

Now, clearly,

$$
\begin{equation*}
\int_{T_{0}} d^{3} \mathbf{v} \frac{v_{1}^{2}}{\mathbf{v}^{2}} Q(\mathbf{v})=\int_{T_{0}} d^{3} \mathbf{v} \frac{v_{2}^{2}}{\mathbf{v}^{2}} Q(\mathbf{v})=\int_{T_{0}} d^{3} \mathbf{v} \frac{v_{3}^{2}}{\mathbf{v}^{2}} Q(\mathbf{v}) \tag{3.35}
\end{equation*}
$$

and therefore,

$$
\begin{equation*}
\frac{2}{\pi^{2}} \int_{T_{0}} d^{3} \mathbf{v} \frac{v_{1}^{2}}{\mathbf{v}^{2}} Q(\mathbf{v})=\frac{1}{3} \frac{2}{\pi^{2}} \int_{T_{0}} d^{3} \mathbf{v} \frac{v_{1}^{2}+v_{2}^{2}+v_{3}^{2}}{\mathbf{v}^{2}} Q(\mathbf{v})=\frac{2}{3 \pi^{2}} \int_{T_{0}} d^{3} \mathbf{v} Q(\mathbf{v}) \tag{3.36}
\end{equation*}
$$

Hence, we have shown that

$$
\begin{equation*}
\int_{T_{0}} d^{3} \mathbf{v} Q(\mathbf{v})=\left[\int_{T_{0}} d v \frac{\sin ((2 N+1) v)}{\sin v}\right]^{3}=\pi^{3} \tag{3.37}
\end{equation*}
$$

With the previous relations, we have

$$
\begin{equation*}
\frac{2}{\pi^{2}} \int_{T_{0}} d^{3} \mathbf{v} \frac{(\mathbf{r} \cdot \mathbf{v})^{2}}{\mathbf{v}^{2}} Q(\mathbf{v})=\frac{2 \pi}{3} \mathbf{r}^{2} \tag{3.38}
\end{equation*}
$$

and so we have

$$
\begin{equation*}
I_{\mathbf{0}}\left(\mathbf{r}, \mathfrak{P}_{N}\right)=I_{\mathbf{0}}\left(\mathbf{0}, \mathfrak{P}_{N}\right)-\frac{2 \pi}{3} \mathbf{r}^{2}+O\left((2 N+1)^{-2}\right) \tag{3.39}
\end{equation*}
$$

Combining these results, we have that the potential at the origin due to a particle at $\mathbf{r}$ is

$$
\begin{align*}
& \sum_{\mathbf{n} \in \mathfrak{P}_{N}} \frac{1}{|\mathbf{r}+\mathbf{n}|}=I_{\mathbf{0}}\left(\mathbf{0}, \mathfrak{P}_{N}\right)-\frac{2 \pi}{3} \mathbf{r}^{2}+O\left((2 N+1)^{-2}\right) \\
& \quad+\left[f(\mathbf{r}, \alpha)+\sum_{\substack{\mathbf{m} \in \mathfrak{R} \\
\mathbf{m} \neq \mathbf{0}}} \frac{1}{\pi \mathbf{m}^{2}} \mathbf{e}^{-\pi^{2} \mathbf{m}^{2} / \alpha^{2}+2 \pi i \mathbf{m} \cdot \mathbf{r}}+\sum_{\substack{\mathbf{n} \in \mathfrak{P}_{N} \\
\mathbf{n} \neq \mathbf{0}}} \frac{\operatorname{erfc}(\alpha|\mathbf{r}+\mathbf{n}|)}{|\mathbf{r}+\mathbf{n}|}\right] . \tag{3.40}
\end{align*}
$$

Normally, the charge neutrality condition is invoked to ensure that $I_{0}\left(\mathbf{0}, \mathfrak{P}_{N}\right)$ vanishes when summed over a net neutral set of particles; however, since this term is independent of $\mathbf{r}$, when taking the derivative of the potential to obtain the acceleration this term vanishes without appealing to charge neutrality. Summing this expression over the set of particles and taking the gradient, we have the Redlack-Grindlay result that

$$
\begin{equation*}
\Omega(\mathbf{r})=\Psi(\mathbf{r})-\frac{4 \pi}{3}\left[\sum_{j} m_{j} \mathbf{r}_{j}-\mathbf{r} \sum_{j} m_{j}\right] \tag{3.41}
\end{equation*}
$$

This establishes the relationship between infinite direct summation, $\Psi(\mathbf{r})$, and Ewald summation, $\Omega(\mathbf{r})$; this is precisely what was needed to compute periodic accelerations via direct summation. We now have a solution for periodic boundary conditions which does not modify the local interaction potential. In a sense we have completed the programme begun by Bouchet and Hernquist (1988) and Hernquist et al. (1991) by computing an exact and analytic periodic correction to the equivalent isolated open boundary problem. The only aspect of this theory which remains to be elucidated computationally is that of the infinite sum,

$$
\begin{equation*}
\Psi(\mathbf{r})=\sum_{j=1}^{N} \frac{m_{j}\left(\mathbf{r}_{j}-\mathbf{r}\right)}{\left|\mathbf{r}_{j}-\mathbf{r}\right|^{3}}+\sum_{0<|\mathbf{n}|^{2} \leq 1} \sum_{j=1}^{N} \frac{m_{j}\left(\mathbf{r}_{j}-\mathbf{n}-\mathbf{r}\right)}{\left|\mathbf{r}_{j}-\mathbf{n}-\mathbf{r}\right|^{3}} \tag{3.42}
\end{equation*}
$$

We will show in the next chapter how this infinite sum may be computed at negligible asymptotic cost, thereby explicitly demonstrating that periodicity requires no additional work in comparison to it's open boundary counterpart.

## CHAPTER 4

## Multipole Theory of Infinite Lattice Sums

In the previous chapter, we established that the periodic gravitational acceleration could be computed as a direct sum over replicas. In this chapter we present a simplification of the Fast Multipole Method (FMM) that utilizes only a single grid, rather than the more general hierarchy of grids. FMM is an example of a fast-summation algorithm, which reduces the cost of the N -body problem from $O\left(N^{2}\right)$ to $O(N \log N)$ operations. In this chapter we show how the sum over the infinite replicas may be incorporated into an open boundary solution technique in a straightforward manner at essentially no computational cost.

The speedup exhibited by fast-summation algorithms can be explained by considering sums of the form (c.f. Beatson and Greengard 1997)

$$
\begin{equation*}
u(\mathbf{x})=\sum_{i=1}^{N} m_{i} K\left(\mathbf{x}, \mathbf{x}_{i}\right) \tag{4.1}
\end{equation*}
$$

where, for computing the gravitational potential, $K\left(\mathbf{x}_{i}, \mathbf{x}_{j}\right)=\left|\mathbf{x}_{i}-\mathbf{x}_{j}\right|^{-1}$.
The FMM relies on the notion of a multipole expansion, illustrated as follows. If the kernel $K(\mathbf{x}, \mathbf{y})$ can be expressed as a finite series of order $p$,

$$
\begin{equation*}
K(\mathbf{x}, \mathbf{y}) \approx \sum_{k=1}^{p} \phi_{k}(\mathbf{x}) \psi_{k}(\mathbf{y}) \tag{4.2}
\end{equation*}
$$

it is amenable to a multipole expansion. One first computes the moments

$$
\begin{equation*}
A_{k}=\sum_{i=1}^{N} m_{i} \psi_{k}\left(\mathbf{y}_{i}\right) \tag{4.3}
\end{equation*}
$$

With these, one can then evaluate the desired result as

$$
\begin{equation*}
u(\mathbf{x}) \approx \sum_{k=1}^{p} A_{k} \phi_{k}(\mathbf{x}) \tag{4.4}
\end{equation*}
$$



Figure 4.1: The arrangement of cells in the infinite lattice of replicas. This is a slice through the center of the cubical simulation volume. The black square at the center represents the simulation volume itself; the other squares are its periodic replicas. The white squares immediately adjacent to the center black square are a cross section through the $(2 \mathcal{L}+1)^{3}-1$ nearest neighboring replicas (illustrated are the 26 nearest neighbors corresponding to $\mathcal{L}=1$ ).

The computation is now $O(N p)$; since $p \ll N$ (typically a number of order 10), this is a dramatic increase in computational efficiency in comparison to $O\left(N^{2}\right)$.

In the previous chapter we established the relationship between the acceleration computed by infinite direct summation, $\Psi(\mathbf{r})$, and by Ewald summation, $\Omega(\mathbf{r})$. Denote the partial lattice sums over spherical shells from radius $a$ to $b$ by

$$
\begin{equation*}
\pi(\mathbf{r}, a, b)=\sum_{C(\mathbf{n}, a, b)} \sum_{j=1}^{N} \frac{m_{j}\left(\mathbf{r}_{j}-\mathbf{n}-\mathbf{r}\right)}{\left|\mathbf{r}_{j}-\mathbf{n}-\mathbf{r}\right|^{3}} \tag{4.5}
\end{equation*}
$$

where

$$
\begin{gather*}
C(\mathbf{n}, a, b)=\left\{\mathbf{n} \cdot C_{1}(\mathbf{n}, a) \wedge C_{2}(\mathbf{n}, b)\right\}, \quad \text { with } \\
C_{1}(\mathbf{n}, a)=\left(\left(n_{1} \geq a\right) \vee\left(n_{2} \geq a\right) \vee\left(n_{3} \geq a\right)\right)  \tag{4.6}\\
C_{2}(\mathbf{n}, b)=\left(\left(n_{1} \leq b\right) \wedge\left(n_{2} \leq b\right) \wedge\left(n_{3} \leq b\right)\right)
\end{gather*}
$$

With the assumption that the simulation volume is unity and centered on the origin, the sum over replicas in the periodic acceleration can be partitioned as follows (and illustrated in Figure 4.1),

$$
\begin{equation*}
\Psi(\mathbf{r})=\sum_{j=1}^{N} \frac{m_{j}\left(\mathbf{r}_{j}-\mathbf{r}\right)}{\left|\mathbf{r}_{j}-\mathbf{r}\right|^{3}}+\pi(\mathbf{r}, 1, \mathcal{L})+\pi(\mathbf{r}, \mathcal{L}+1, \infty) \tag{4.7}
\end{equation*}
$$

A closed-form analytic expression (to arbitrary accuracy) for the acceleration due to all infinite replicas excluding the $(2 \mathcal{L}+1)^{3}-1$ nearest neighbours and the simulation volume itself, $\pi(\mathbf{r}, \mathcal{L}+1, \infty)$, can be derived with the machinery of the FMM. The notion of well separatedness of a multipole expansion is equivalent to the concept of the radius of convergence of a power series (a closed domain in which the series will converge). A mathematical criterion which is sufficient to ensure uniform convergence of the multipole series is illustrated in Figure (4.2). The conditions for a multipole expansion can be stated geometrically as follows: a set of source particles at positions $\mathbf{r}_{i}$, such that the $\mathbf{r}_{i}$ may be enclosed within a spherical volume of radius $r_{B}$, is well separated from a position $\mathbf{x}$ within a spherical volume of radius $r_{A}$ enclosing the origin with $\left|\mathbf{r}_{i}\right|>|\mathbf{x}|$, provided that the separation $\mathbf{R}$ between the centers of spheres is greater than the sum of radii, i.e $|\mathbf{R}|>r_{A}+r_{B}$ (i.e. the spheres do not intersect). Under the proviso that these conditions are satisfied, the acceleration at $\mathbf{x}=\left(x_{1}, x_{2}, x_{3}\right)$ may be written using a Cartesian multipole
expansion as

$$
\sum_{j=1}^{N} \frac{m_{j}\left(\mathbf{r}_{j}-\mathbf{x}\right)}{\left|\mathbf{r}_{j}-\mathbf{x}\right|^{3}}=\sum_{p=0}^{\infty} \sum_{\alpha=0}^{p} \sum_{\beta=0}^{p-\alpha} \sum_{\gamma=0}^{p-\alpha-\beta} x_{1}^{\alpha} x_{2}^{\beta} x_{3}^{\gamma} \times\left(\begin{array}{c}
(\alpha+1) T^{(\alpha+1), \beta, \gamma}  \tag{4.8}\\
(\beta+1) T^{\alpha,(\beta+1), \gamma} \\
(\gamma+1) T^{\alpha, \beta,(\gamma+1)}
\end{array}\right)
$$

In writing this expression, we have used a Cartesian multipole expansion based upon the formulation of Hinsen and Felderhof (1992), where, for $|\mathbf{r}|<\left|\mathbf{r}^{\prime}\right|$,

$$
\begin{equation*}
\frac{1}{\left|\mathbf{r}-\mathbf{r}^{\prime}\right|}=\sum_{p=0}^{\infty} \sum_{\alpha=0}^{p} \sum_{\beta=0}^{p-\alpha} \sum_{\gamma=0}^{p-\alpha-\beta} \mathcal{M}^{\alpha \beta \gamma}(\mathbf{r}) \mathcal{D}^{\alpha \beta \gamma}\left(\mathbf{r}^{\prime}\right)=\sum_{\substack{\alpha \beta \gamma \\ p=0}}^{\infty} \mathcal{M}^{\alpha \beta \gamma}(\mathbf{r}) \mathcal{D}^{\alpha \beta \gamma}\left(\mathbf{r}^{\prime}\right) \tag{4.9}
\end{equation*}
$$

where the second equality introduces a more compact notation for summing over the multipole orders; for each value of $p$, the sum is taken over all values of $\alpha, \beta$, and $\gamma$ such that $\alpha+\beta+\gamma=p$. When the sum over $p$ is truncated at $p=P$, this is referred to as an "order $P$ multipole expansion." The expressions for $\mathcal{D}$ and $\mathfrak{M}$ are defined as

$$
\begin{align*}
\mathcal{D}^{\alpha \beta \gamma}(\mathbf{r}) & =\frac{\partial^{\alpha+\beta+\gamma}}{\partial r_{x}^{\alpha} \partial r_{y}^{\beta} \partial r_{z}^{\gamma}}\left(\frac{1}{|\mathbf{r}|}\right) \\
\mathcal{M}^{\alpha \beta \gamma}(\mathbf{r}) & =\frac{|\mathbf{r}|^{2(\alpha+\beta+\gamma)+1}}{(2(\alpha+\beta+\delta)-1)!!\alpha!\beta!\gamma!} D^{\alpha \beta \gamma}(\mathbf{r}) \tag{4.10}
\end{align*}
$$

Using this formalism, the Taylor coefficients, $T$ are represented as

$$
\begin{equation*}
T^{\alpha \beta \gamma}=\frac{(-1)^{(\alpha+\beta+\gamma)}}{\alpha!\beta!\gamma!} \sum_{\substack{\delta \varepsilon \zeta \\ p=0}}^{\infty} \Delta^{\delta \varepsilon \zeta} \mathcal{D}^{(\alpha+\delta)(\beta+\varepsilon)(\gamma+\zeta)}(\mathbf{R}) \tag{4.11}
\end{equation*}
$$

with the multipole moments

$$
\begin{equation*}
\Delta^{\delta \varepsilon \zeta}=\sum_{j=1}^{N} m_{i} \mathcal{M}^{\delta \varepsilon \zeta}\left(\mathbf{r}_{i}-\mathbf{R}\right) \tag{4.12}
\end{equation*}
$$



Figure 4.2: Geometry of a well-separated multipole interaction: the acceleration at $\vec{x}$ due to particles $\vec{r}_{i}$ may be obtained provided $\left|\vec{r}_{i}\right|>|\vec{x}|$ and $r_{A}+r_{B}<R$ where $|\vec{x}|<r_{A}$ and $\left|\vec{\delta}_{i}\right|=\left|\vec{r}_{i}-\vec{R}\right|<r_{B}$; the origin is denoted by the shaded gray circle.

This formalism permits us to express the acceleration at position $\mathbf{r}=\left(r_{1}, r_{2}, r_{3}\right)$, due to all replicas except for the simulation volume itself and it's $(2 \mathcal{L}+1)^{3}-1$ nearest neighbours as

$$
\pi(\mathbf{r}, \mathcal{L}+1, \infty)=\sum_{\substack{\alpha \beta \gamma  \tag{4.13}\\
p=0}}^{\infty} r_{1}^{\alpha} r_{2}^{\beta} r_{3}^{\gamma} \times\left(\begin{array}{c}
(\alpha+1) \Gamma^{(\alpha+1), \beta, \gamma} \\
(\beta+1) \Gamma^{\alpha,(\beta+1), \gamma} \\
(\gamma+1) \Gamma^{\alpha, \beta,(\gamma+1)}
\end{array}\right)
$$

where

$$
\begin{equation*}
\Gamma^{i j k}=\frac{-1^{i+j+k}}{i!j!k!} \sum_{\substack{\delta \varepsilon \zeta \\ p=0}}^{\infty} \frac{\Delta^{\delta, \varepsilon, \zeta}}{\delta!\varepsilon!\zeta!} \Theta^{i+\delta, j+\varepsilon, k+\zeta} \tag{4.14}
\end{equation*}
$$

with

$$
\begin{equation*}
\Theta^{i j k}=\sum_{C_{1}(\mathbf{n}, \mathcal{L}+1)} D^{i j k}(\mathbf{n}) \tag{4.15}
\end{equation*}
$$

and it is understood that these sums in $\Theta^{i j k}$ are accumulated over spherical shells. This representation illustrates the crucial aspect of our formalism; namely that any reference to the mass distribution can be removed from the lattice summation, and only $\Theta^{i, j, k}$ contains any dependence on the lattice. Because of this, the $\Theta^{i, j, k}$ may be precomputed once and for all.

In spite of the Cartesian appearance of the tensors in the derivation above, $\Theta^{\alpha \beta \gamma}, \Delta^{\alpha \beta \gamma}$ and $T^{\alpha \beta \gamma}$ are all trace-free; that is, they satisfy the following relation, for $\gamma>1$,

$$
\begin{equation*}
\zeta^{\alpha \beta \gamma}=-\zeta^{(\alpha+2) \beta(\gamma-2)}-\zeta^{\alpha(\beta+2)(\gamma-2)} \tag{4.16}
\end{equation*}
$$

The computation of the derivatives, $\Theta^{\alpha \beta \gamma}$, proceeds directly from repeated application of the chain-rule; one obtains the relations

$$
\begin{align*}
& D^{(0,0,0)} \phi(\mathbf{x})=|\mathbf{x}|^{-1} \\
& D^{(1,0,0)} \phi(\mathbf{x})=-x_{1}|\mathbf{x}|^{-3} \quad D^{(0,1,0)} \phi(\mathbf{x})=-x_{2}|\mathbf{x}|^{-3} \quad D^{(0,0,1)} \phi(\mathbf{x})=-x_{3}|\mathbf{x}|^{-3} \\
& D^{(1,1,0)} \phi(\mathbf{x})=3 x_{1} x_{2}|\mathbf{x}|^{-5} \quad D^{(1,0,1)} \phi(\mathbf{x})=3 x_{1} x_{3}|\mathbf{x}|^{-5} \quad D^{(0,1,1)} \phi(\mathbf{x})=3 x_{2} x_{3}|\mathbf{x}|^{-5} \\
& D^{(1,1,1)} \phi(\mathbf{x})=-15 x_{1} x_{2} x_{3}|\mathbf{x}|^{-7} \tag{4.17}
\end{align*}
$$

and for $n_{1} \geq 2$,

$$
\begin{align*}
& \mathcal{D}^{\left(n_{1}, 0,0\right)}=-|\mathbf{x}|^{2}\left(\left(2 n_{1}-1\right) x_{1} \mathcal{D}^{\left(n_{1}-1,0,0\right)}+\left(n_{1}-1\right)^{2} \mathcal{D}^{\left(n_{1}-2,0,0\right)}\right) \\
& \mathcal{D}^{\left(n_{1}, 1,0\right)}=-|\mathbf{x}|^{2}\left(\left(2 n_{1}-1\right) x_{1} \mathcal{D}^{\left(n_{1}-1,1,0\right)}+\left(n_{1}-1\right)^{2} \mathcal{D}^{\left(n_{1}-2,1,0\right)}+2 x_{2} \mathcal{D}^{\left(n_{1}, 0,0\right)}\right) \\
& \mathcal{D}^{\left(n_{1}, 0,1\right)}=-|\mathbf{x}|^{2}\left(\left(2 n_{1}-1\right) x_{1} \mathcal{D}^{\left(n_{1}-1,0,1\right)}+\left(n_{1}-1\right)^{2} \mathcal{D}^{\left(n_{1}-2,0,1\right)}+2 x_{3} \mathcal{D}^{\left(n_{1}, 0,0\right)}\right)  \tag{4.18}\\
& \mathcal{D}^{\left(n_{1}, 1,1\right)}=-|\mathbf{x}|^{2}\left(\left(2 n_{1}-1\right) x_{1} \mathcal{D}^{\left(n_{1}-1,1,1\right)}+\left(n_{1}-1\right)^{2} \mathcal{D}^{\left(n_{1}-2,1,1\right)}\right. \\
& \\
& \left.\quad+2 x_{2} \mathcal{D}^{\left(n_{1}, 0,1\right)}+2 x_{3} \mathcal{D}^{\left(n_{1}, 1,0\right)}\right)
\end{align*}
$$

and similarly for $\mathcal{D}^{\left(0, n_{2}, 0\right)}, \mathcal{D}^{\left(1, n_{2}, 0\right)}, \mathcal{D}^{\left(0, n_{2}, 1\right)}, \mathcal{D}^{\left(1, n_{2}, 1\right)}$. Now to use the simplified recur-
sion which exploits the traceless nature of the tensor, we require two last expressions,

$$
\begin{align*}
\mathcal{D}^{\left(n_{1}, n_{2}, 0\right)}=-|\mathbf{x}|^{2}[ & \left(2 n_{1}-1\right) x_{1} \mathcal{D}^{\left(n_{1}-1, n_{2}, 0\right)}+\left(n_{1}-1\right)^{2} \mathcal{D}^{\left(n_{1}-2, n_{2}, 0\right)}  \tag{4.19}\\
& \left.+2 n_{2} x_{2} \mathcal{D}^{\left(n_{1}, n_{2}-1,0\right)}+n_{2}\left(n_{2}-1\right) \mathcal{D}^{\left(n_{1}, n_{2}-2,0\right)}\right]
\end{align*}
$$

and

$$
\begin{align*}
\mathcal{D}^{\left(n_{1}, n_{2}, 1\right)}= & -|\mathbf{x}|^{2}\left(\left(2 n_{1}-1\right) x_{1} \mathcal{D}^{\left(n_{1}-1, n_{2}, 1\right)}+\left(n_{1}-1\right)^{2} \mathcal{D}^{\left(n_{1}-2, n_{2}, 1\right)}\right.  \tag{4.20}\\
& \left.+2 n_{2} x_{2} \mathcal{D}^{\left(n_{1}, n_{2}-1,1\right)}+n_{2}\left(n_{2}-1\right) \mathcal{D}^{\left(n_{1}, n_{2}-2,1\right)}+2 x_{3} \mathcal{D}^{\left(n_{1}, n_{2}, 0\right)}\right)
\end{align*}
$$

$\Theta^{i j k}$ will be identically zero if any if any one of the $\alpha, \beta, \gamma$ is an odd number; this can be seen by considering the algebraic structure of the derivatives (notably the odd powers of the variables $x_{i}$ ) and the set

$$
\begin{array}{r}
\{\{a, b, c\},\{a, b,-c\},\{a,-b, c\},\{a,-b,-c\},\{-a, b, c\},  \tag{4.21}\\
\{-a, b,-c\},\{-a,-b, c\},\{-a,-b,-c\}\}
\end{array}
$$

Exploiting the cubic symmetry of the lattice sum, it is, further, only necessary to consider values of the lattice vector $\mathbf{n}$ in the first octant, as

$$
\begin{equation*}
\sum_{C_{1}(\mathbf{n}, R+1)} \mathcal{D}^{\alpha \beta \gamma}(\mathbf{n})=\sum_{\substack{C_{1}(\mathbf{n}, R+1) \\ n_{1}>0, n_{2}>0, n_{3}>0}} \delta\left(n_{1}, n_{2}, n_{3}\right) \mathcal{D}^{\alpha \beta \gamma}(\mathbf{n}) \tag{4.22}
\end{equation*}
$$

where $\delta\left(n_{1}, n_{2}, n_{3}\right)=(8,4,2,1)$ if three, two, one, or none of $n_{1}, n_{2}, n_{3}$ are non-zero, respectively. Together with the trace-tree relation, this implies that one need only evaluate $\Theta^{2 \alpha, 2 \beta, 0}$ for $\alpha, \beta \leq p / 2$; the rest of the components may be computed by recursion. In practice, of course, one must truncate the infinite sums in $\Theta^{2 \alpha, 2 \beta, 0}$, i.e. the sum over periodic replicas in $\Theta$ is truncated at a radius $|\mathbf{n}|=\mathcal{L}_{\text {max }}$ from the simulation volume. In order to perform this expansion, we use quad-double precision arithmetic ( 64 significant
digits) ${ }^{1}$. The upper bound for the sum was truncated at $|\mathbf{n}|^{2}<3 \times 200^{2}$, where all the $\mathcal{D}^{2 \alpha, 2 \beta, 0}(\mathbf{n})$ underflow quad double precision. We emphasize that these $\Theta^{2 \alpha, 2 \beta, 0}$ need be computed once and for all (irrespective of the distribution of the particles) and saved to disk. Remarkably, even for an order 16 expansion, there are only 64 scalars required to represent the effect of the infinite lattice.

The expression for $\mathcal{M}^{\alpha \beta \gamma}\left(\mathbf{r}_{i}\right)$ contains the factors $\left|\mathbf{r}_{i}\right|^{2(\alpha+\beta+\gamma)+1} \mathcal{D}^{\alpha \beta \gamma}\left(\mathbf{r}_{i}\right)$. Clearly, if one had to form multipoles from every point in the simulation using this explicit expression, the cost of computing a derivative for every particle in the system would be staggeringly expensive. Instead, we use a result from Cipriani (1982),

$$
\begin{equation*}
\frac{\partial^{\alpha}}{\partial x^{\alpha}}\left(\frac{1}{|\mathbf{r}|}\right)=\sum_{s=0}^{\lfloor\alpha / 2\rfloor} a_{s}^{\alpha} x^{\alpha-2 s}|\mathbf{r}|^{2 s-2 \alpha-1} \tag{4.23}
\end{equation*}
$$

where

$$
\begin{equation*}
a_{s}^{\alpha}=\frac{-1^{\alpha+s}(2 \alpha-2 s-1)!!\alpha!}{2^{s} s!(\alpha-2 s)!} \tag{4.24}
\end{equation*}
$$

Using this result, we can write a much simpler polynomial expression for $\widetilde{M}^{\alpha \beta \gamma}(\mathbf{r})$. With $\mathbf{r}=(x, y, z)$,

$$
\begin{equation*}
|\mathbf{r}|^{2(\alpha+\beta+\gamma)+1} D^{\alpha \beta \gamma}(\mathbf{r})=\sum_{s=0}^{\lfloor\alpha / 2\rfloor} \sum_{t=0}^{\lfloor\beta / 2\rfloor} \sum_{u=0}^{\lfloor\gamma / 2\rfloor} a_{s t u}^{\alpha \beta \gamma} x^{\alpha-2 s} y^{\beta-2 t} z^{\gamma-2 u} \tag{4.25}
\end{equation*}
$$

where

$$
\begin{equation*}
a_{s t u}^{\alpha \beta \gamma}=\frac{-1^{s+t+u}(2(\alpha+\beta+\gamma)-2 s-2 t-2 u)!}{s!t!u!(\alpha-2 s)!(\beta-2 t)!(\gamma-2 u)!(\alpha+\beta+\gamma-s-t-u)!} \tag{4.26}
\end{equation*}
$$

Since this is a polynomial in $x^{a} y^{b} z^{c}$, we can form the full Cartesian multipoles,

$$
\begin{equation*}
Q^{\alpha \beta \gamma}\left(\mathbf{r}_{l}\right)=m_{l} r_{l, 1}^{\alpha} r_{l, 2}^{\beta} r_{l, 3}^{\gamma} \tag{4.27}
\end{equation*}
$$

[^2]and then express the reduced multipoles in terms of the full Cartesian multipoles as the linear sum,
\[

$$
\begin{equation*}
m_{l}\left|\mathbf{r}_{l}\right|^{2(\alpha+\beta+\gamma)+1} \mathcal{D}^{\alpha \beta \gamma}\left(\mathbf{r}_{l}\right)=\sum_{\delta+\varepsilon+\zeta=l} q_{\delta \varepsilon \zeta} Q^{\delta \varepsilon \zeta}\left(\mathbf{r}_{l}\right) \tag{4.28}
\end{equation*}
$$

\]

The coefficients $q_{\delta \varepsilon \zeta}$ are pre-computed using the Mathematica script in Appendix (A). Again, the trace-free relation reduces the space required to store these trace-free tensors; only $(P+1)^{2}$ elements need be stored and computed explicitly rather than the full Cartesian set of $(P+1)(P+2)(P+3) / 6$ elements. The remaining Cartesian elements may rapidly be determined, as needed, by recursion. This obviates what has traditionally been the main objection to using Cartesian tensors over their spherical counterparts, namely that Cartesian tensors require more memory to store them. Recall that the spherical harmonic representation of the potential at $\mathbf{r}, \Phi(\mathbf{r})$, due to a set of $N$ source particles with masses $\left\{m_{i}\right\}$ and positions $\left\{\left(r_{i}, \theta_{i}, \phi_{i}\right)\right\}$ is

$$
\begin{equation*}
\Phi(\mathbf{r})=\sum_{l=0}^{\infty} \sum_{m=-1}^{l} \frac{\alpha_{l}^{m}}{r^{l+1}} Y_{l}^{m}(\theta, \phi) \tag{4.29}
\end{equation*}
$$

where the spherical harmonics are expressed in terms of the Legendre polynomials $P_{l}^{m}(x)$,

$$
\begin{equation*}
Y_{l}^{m}(\theta, \phi)=\sqrt{\frac{2 l+1}{4 \pi} \frac{(l-m)!}{(l+m)!}} P_{l}^{m}(\cos \theta) e^{i m \phi} \tag{4.30}
\end{equation*}
$$

and the spherical harmonic multipole moments are

$$
\begin{equation*}
\alpha_{l}^{m}=\frac{4 \pi}{2 l+1} \sum_{i=1}^{N} m_{i} r_{i}^{l} Y_{l}^{m *}\left(\theta_{i}, \phi_{i}\right) \tag{4.31}
\end{equation*}
$$

The asterisk (*) denotes complex conjugation. In addition to using the same storage space as spherical multipoles, working in the reduced Cartesian representation is computationally much faster; it does not require the evaluation of special functions.

The only question which remains in our formulation concerns the determination of $\mathcal{L}$, with the obvious requirement that $\mathcal{L} \geq 1$; i.e. at least the 26 adjacent neighbouring replicas
in the lattice must be in the simulation volume's near field. Our intention is to compute $\pi(r, \mathcal{L}+1, \infty)$ to machine precision, with the constraint that we have only a modest range in $p$, say $p \leq 16$, at our disposal. Consider Figure 4.2, the absolute error bound for the acceleration derived by Salmon and Warren (1994) for a Barnes-Hut multipole expansion,

$$
\begin{equation*}
\delta a_{(p)}(\vec{x})=\left(\frac{1}{d-b}\right)^{2}\left((p+2)\left(\frac{\left\lceil B_{(p+1)}\right\rceil}{d^{p+1}}\right)-(p+1)\left(\frac{\left\lfloor B_{(p+2)}\right\rfloor}{d^{p+2}}\right)\right) \tag{4.32}
\end{equation*}
$$

where $d=|\vec{x}-\vec{R}|$ and without loss of generality we assume $\vec{R}$ is the center of gravity of $\vec{r}_{i}$ and

$$
\begin{equation*}
b=\max _{i}\left|\vec{r}_{i}-\vec{R}\right|, \quad B_{(n)}=\sum_{i} m_{i}\left|\vec{r}_{i}-\vec{R}\right|^{n}, \tag{4.33}
\end{equation*}
$$

This analytic bound for the error in the acceleration is, unfortunately, too weak to be of practical guidance. An illustration of the actual expected maximum relative errors in the acceleration for the Barnes-Hut expansion,

$$
\begin{equation*}
\phi(\vec{x})=\sum_{\substack{\alpha \beta \gamma \\ p=0}}^{\infty} \Delta^{\alpha \beta \gamma}(\delta) \mathcal{D}^{\alpha \beta \gamma}(\vec{R}-\vec{x}) \tag{4.34}
\end{equation*}
$$

can examined by Monte Carlo sampling. We consider two source distributions within a unit volume; a single particle located at $(0.5,0.5,0.5)$, and a quasi-uniformly distributed set of 1024 particles. Naturally a large number of source particles distributed over the region will produce a smoother acceleration field, and therefore lead to smaller relative errors, than a single particle. The results are illustrated in Figures 4.3 and 4.4. Note that for the fiducial selection of a separation of $|\vec{R}|=2$ and $p=16$, the relative error distribution for the single particle test is tightly clustered for both slices at $x=1.5$ and $x=2.5$.

In contrast, consider the distribution of errors in the Fast Multipole Method expansion,

$$
\begin{equation*}
\phi(\vec{x})=\sum_{\substack{\alpha \beta \gamma \\ p=0}}^{\infty} \frac{-1^{\alpha+\beta+\gamma}}{\alpha!\beta!\gamma!} x_{1}^{\alpha} x_{2}^{\beta} x_{3}^{\gamma} \sum_{\substack{\delta \varepsilon \zeta \\ p=0}}^{\infty} \Delta^{\alpha \beta \gamma}(\delta) \mathcal{D}^{\alpha+\delta, \beta+\varepsilon, \gamma+\zeta}(\vec{R}) \tag{4.35}
\end{equation*}
$$

This is illustrated in Figures 4.5 and 4.6. Not unexpectedly, one observes that the double multipole expansion (FMM) incurs a greater penalty in relative error than it's single multipole expansion counterpart (BH). A particularly curious artifact of the FMM expansion is that its range in relative error is much larger than the quite narrow range of the BH expansion. The BH relative error distribution exhibits a property which is reminiscent of Chebyshev economization; the maximum error is reduced at the expense of increasing the error in certain other parts of the domain. The upper bound error estimate in the acceleration for the FMM interaction derived by (Dehnen, 2002),

$$
\begin{equation*}
A_{p}<(p+1)\left(\frac{r_{A}+r_{B}}{R}\right)^{p}\left[\frac{1}{R-\left(r_{A}+r_{B}\right)}\right]^{2} \tag{4.36}
\end{equation*}
$$

is plagued by the same malady as the BH analytic estimate of Salmon and Warren (1994); it is too weak.

The intricacies of error bounds aside, we note that the maximum relative errors in Figures 4.5 and 4.6 should not be construed as the relative error in the total acceleration. These are merely the relative errors in the particle accelerations due only to the source particles considered. In practice, the total acceleration will be dominated by the contribution from the acceleration due to nearby particles and will be much larger than any far field partial acceleration. This results in a total relative error in the acceleration which is much smaller than that of the acceleration due to the far field alone.

Having outlined the evaluation of $\pi(\mathbf{r}, \mathcal{L}+1, \infty)$, the remaining sum,

$$
\begin{equation*}
\pi(\mathbf{r}, 0,0)+\pi(r, 1, \mathcal{L}) \tag{4.37}
\end{equation*}
$$

can be evaluated in any manner one desires,

$$
\begin{equation*}
\text { Acceleration }(\mathbf{r}, T)+\sum_{C(\mathbf{n}, 1, L)} \text { Acceleration }(\mathbf{r}+\mathbf{n}, T) \tag{4.38}
\end{equation*}
$$



Figure 4.3: The left and right plots illustrate respectively the maximum relative errors in a Barnes Hut expansion for single source particle at $(0.5,0.5,0.5)$ and a uniform distribution of 1024 source particles at varying separations of $|\vec{R}|$ and orders $p=2,4,8,16$ (from the top of the graph) as computed by Monte Carlo sampling.


Figure 4.4: The left and right panels show the distribution of relative errors in the acceleration for a BH expansion at a separation of $\vec{R}=2$ and order $p=16$ for the single source particle in two $y-z$ planes, $z=1.5$ and $z=2.5$ respectively.


Figure 4.5: The left and right plots illustrate respectively the maximum relative errors in a FMM expansion for single source particle at $(0.5,0.5,0.5)$ and a uniform distribution of 1024 source particles at varying separations of $|\vec{R}|$ and orders $p=2,4,8,16$ (from the top of the graph) as computed by Monte Carlo sampling.


Figure 4.6: The left and right panels show, respectively, the distribution of relative errors in the acceleration for a FMM expansion at a separation of $\vec{R}=2$ and order $p=16$ for the single source particle in two $y-z$ planes, $z=1.5$ and $z=2.5$ respectively.
where Acceleration $(\mathbf{x}, T)$ denotes the evaluation of the acceleration at position $\mathbf{x}$ using method $T . T$ could be the ubiquitous monopole Barnes Hut tree, a high-order FMM tree, or even a direct calculation if the number of particles is not prohibitively large. The monopole BH tree typically yields a $99 \%$ percentile in the relative error in the acceleration of a few percent. As our goal is to obtain accelerations accurate to machine precision, the monopole BH tree would require an excessive number of interactions; instead, we use high-order (order 8 or 16) trees which we discuss in the next chapter.

We conclude this chapter with an illustration of the accuracy this representation can achieve. Using our direct summation over replicas formalism, the periodic acceleration due to $32^{3}$ particles was computed and compared with the actual Ewald acceleration formula; naturally, in order to compute the Ewald acceleration from it's canonical representation, even for $32^{3}$ particles this was computationally very expensive. The distribution was obtained from the initial conditions code described in Sirko (2005) from standard LCDM parameters. The initial separations of particles in this distribution is such that the use of force softening may be omitted. The reference "exact" computation was performed using (1.6), the analytic Ewald acceleration formula, with $|\mathbf{n}| \leq 4$ and $|\mathbf{h}| \leq 4$, using quaddouble precision. Choosing a multipole order $P=16$ and a separation distance $\mathcal{L}$ of one cell yields a maximum relative error in the acceleration of $5 \times 10^{-6}$. Choosing multipole order $P=16$ and a separation $\mathcal{L}=2$, giving 124 adjacent neighboring cells, our computation agrees with the reference values to 16 significant digits. Using a fast direct summation implementation (using only a 32 bit representation of floating point numbers), the maximum relative error increases, but only to $5 \times 10^{-5}$.

In summary, we have demonstrated that the effect of the infinite lattice sum may be precisely represented (to machine precision) by a tensor product of the multipole coefficients and 64 scalar coefficients (for order 16) whilst retaining a $1 / r$ kernel for the near
field representation. In contrast the method of Hernquist et al. (1991) required tabulating an Ewald correction to the open boundary problem which by its very construction was only very approximate (at best a $99 \%$ relative error in the acceleration of $1 \%$ ). In the next chapter we will outline our method for implementing a high order (order 8 and 16) FMM scheme which will be used to evaluate the near field replicas.

## Chapter 5

## A Pseudo-Particle Fast Multipole Method

In the previous chapter we introduced the mathematical and computational machinery of multipole expansions and its application to infinite lattice sums. However, recently it has been recognized that a purely algebraic implementation of the FMM formulæfor the Taylor series coefficients presented in the previous chapter can be made more computationally efficient; in particular we mention the fast linear algebra implementations of FMM explicitly coded for the GPU by Gumerov and Duraiswami (2008). Instead in this chapter, we develop a new variation of the FMM that uses pseudo-particles (Anderson, 1992; Makino, 1999).

We begin with a qualitative presentation of our method. The standard monopole Barnes Hut ( BH ) tree algorithm replaces the interaction between a sink particle and a well separated group of sources by a single interaction between the center-of-mass of the sources and the sink; this is illustrated in Figure (5.1). In this monopole representation, as the desired accuracy of the approximation is increased, the separation radius which must be enforced increases rapidly. All interactions with particles within this radius must be computed as direct, particle-particle interactions, and so the number of particles with which the sink particle must interact becomes unacceptably large. One can ameliorate this situation by resorting to higher order multipoles. Recalling electrostatics, one can always replace a set of charges within a volume with another set of charges on the surface surrounding that volume such that the multipole moments are equivalent. This is the idea behind the "pseudo-particle" method and is illustrated schematically in Figure (5.2). Our desire for computational efficiency dictates two requirements. First, the number of pseudo-particles on the surface of the sphere must be significantly less than


Figure 5.1: A monopole Barnes-Hut interaction. A set of source particles (black) wellseparated from a sink particle (blue) are collapsed to a single pseudo-particle at their center of mass (red). The effect of the sources on the sink is then computed as a single interaction from the center of mass.
the number within the sphere; otherwise, one might as well have computed the interactions directly. Second, the effort required to determine the positions and masses of these pseudo-particles should not exceed that of doing the direct interactions they replace. A greater economy of direct interactions is achieved by employing FMM instead of BarnesHut. The main contribution of this chapter is to show how to determine a single set of $K$ pseudo-particles surrounding a set of sink particles which represent the effect of the entire far field multipoles of the rest of the well separated sources, as illustrated in Figure (5.3). Instead of trying to obtain the $K$ (black) pseudo particles directly from $M$ sets of pseudoparticles, we first compute the potential at $K$ positions around the sink particles due to the QM source sets and then analytically derive a set of pseudo-particles which would induce this same potential. This is the key operation in this pseudo-particle FMM scheme, and to the best of our knowledge it is the first time it has been derived analytically.

Partitioning space as usual using a tree structure, we represent the multipole moments of the particles in each cell of the tree by a much smaller number of pseudo-particles at a set of fixed positions on a sphere surrounding the cell. The masses of these particles


Figure 5.2: A higher order multipole Barnes-Hut interaction. Instead of replacing the well-separated set of $M$ sink particles with a single pseudo-particle, the set is replaced by $K$ pseudo-particles on the surface of a sphere surrounding the set whenever $K \ll M$.
are determined by a simple matrix multiplication. In order to construct a representation of the acceleration in each cell due to distant cells, we start by computing the Newtonian $1 / r$ potential at a set of points on a sphere surrounding the cell due to the distant pseudo-particles. An intriguing aspect of this calculation is that the potential is a purely additive scalar; hence concerns normally arising relating to the accurate summation of partial accelerations whose magnitudes vary by several orders and posses differing signs are largely obviated; moreover the potential between two particles is typically twice as fast to compute as the corresponding acceleration. The computation of the potential is itself a simple N-body calculation, well-adapted to the GPU. By Laplace's theorem, if we know the potential on this surface, we know it throughout the enclosed volume. We then form another set of pseudo-particles just outside the potential sphere. The acceleration of particles within the sphere can then be computed by yet another small N -body calculation. Figure (5.4) illustrates all of the machinery needed to implement the FMM scheme using a purely particle interaction. We first compute, for each cell, the equivalent pseudo-particles to generate the multipoles in the interior of the cell. Then, for each sink cell which is well


Figure 5.3: With $Q$ nodes surrounded by $K$ pseudo-particles, the Barnes-Hut scheme requires $K Q$ interactions to obtain the acceleration each blue filled circle. Instead in the FMM scheme one desires to find a single set of $K$ pseudo-particles (black filled circles ) which represents the sum of these interactions thereby increasing the efficiency of the scheme by a factor of $Q$.
separated, we compute the potential due to all of the source pseudo-particles on the surface of the cell. Once this has been done for all cells which need to interact with one another, we can now compute for every cell the equivalent pseudo-particles to generate that potential for each cell.

All previously-published techniques (e.g. Ying et al., 2004; Chau et al., 2008) to determine the set of pseudo-particle masses from a set of potentials have resorted to techniques from linear algebra (such as Tikhonov regularization) to solve an extremely ill-conditioned linear system. These methods are expensive as well as inaccurate for expansions of high order. In contrast, we have a developed a completely analytic formalism, and can therefore determine the pseudo-particle masses corresponding to a set of potentials to arbitrary accuracy, and with much less computational effort. Using this


Figure 5.4: Particle-based FMM. A set of pseudo-particles is found that matches the high-order multipoles of a set of source particles (1). We then calculate the potential $\Phi$ induced by this set of pseudo-particles at a set of points on a sphere surrounding the sink particles (2). We then find a new set of pseudo-particles on a sphere of larger radius that match the potential on this ring (3). Finally, the interaction between all well-separated source particles and the sink particles is computed from this single set of pseudo-particles (4). The position of all pseudo-particles is known in advance; only the masses vary. Our breakthrough is to be able to use an analytic inversion to compute these masses as a simple multiplication by a fixed matrix. Note that the computation of the potential can be summed from many rings and involves only a simple $1 / r$ calculation.
pseudo-particle technology, we have an FMM where the large majority of the computational work takes the form of direct N -body interactions and thus naturally lends itself to the GPU. Our current implementation of pseudo-particle FMM achieves speeds in excess of 10 times the fast linear algebra implementations of FMM explicitly coded for the GPU by Gumerov and Duraiswami (2008).

The analytic representations we need to develop require the formulation of the multipole expansion in spherical harmonics. We express the potential at $\mathbf{r}, \Phi(\mathbf{r})$, due to a set of $N$ source particles with masses $\left\{m_{i}\right\}$ and positions $\left\{\left(r_{i}, \theta_{i}, \phi_{i}\right)\right\}$ as

$$
\begin{equation*}
\Phi(\mathbf{r})=\sum_{l=0}^{\infty} \sum_{m=-1}^{l} \frac{\alpha_{l}^{m}}{r^{l+1}} Y_{l}^{m}(\theta, \phi) \tag{5.1}
\end{equation*}
$$

where the spherical harmonics are expressed in terms of the Legendre polynomials $P_{l}^{m}(x)$,

$$
\begin{equation*}
Y_{l}^{m}(\theta, \phi)=\sqrt{\frac{2 l+1}{4 \pi} \frac{(l-m)!}{(l+m)!}} P_{l}^{m}(\cos \theta) e^{i m \phi} \tag{5.2}
\end{equation*}
$$

and the spherical harmonic multipole moments are

$$
\begin{equation*}
\alpha_{l}^{m}=\frac{4 \pi}{2 l+1} \sum_{i=1}^{N} m_{i} r_{i}^{l} Y_{l}^{m *}\left(\theta_{i}, \phi_{i}\right) \tag{5.3}
\end{equation*}
$$

The asterisk (*) denotes complex conjugate.
We wish to replace this set of particles with $K$ pseudo-particles with masses $M_{j}$ and positions $\mathbf{R}_{j}=\left(a, \theta_{j}, \phi_{j}\right)$, where $a$ is the radius of the sphere. Equating the $\alpha_{l}^{m}$ of the pseudo-particles and the original set, the pseudo-particle masses must satisfy the relations

$$
\begin{equation*}
\sum_{i=1}^{N} m_{i} r_{i}^{l} Y_{l}^{m *}\left(\theta_{i}, \phi_{i}\right)=\sum_{j=1}^{K} a^{l} M_{j} Y_{l}^{m *}\left(\theta_{j}, \phi_{j}\right) \tag{5.4}
\end{equation*}
$$

for $0 \leq l \leq p$ and $-l \leq m \leq l$. Makino's approach is to fix the positions $\left\{\mathbf{R}_{j}\right\}$ and solve the resulting linear system for the masses $M_{j}$.

The question arises: How many points $K$ are necessary to represent a given multipole order, and what is a good distribution of the $\left\{\mathbf{R}_{j}\right\}$ ? It turns out that an excellent distribution for the $\left\{\mathbf{R}_{j}\right\}$ is a three dimensional spherical t-design (Makino, 1999). A set of $K$ points $X_{K}=\left\{\mathbf{x}_{1}, \cdots, \mathbf{x}_{K}\right\}$ on the unit sphere $S^{2}$ is a three dimensional spherical $t$-design if the identity

$$
\begin{equation*}
\int_{S^{2}} f(\mathbf{x}) d \mu(\mathbf{x})=\frac{1}{K} \sum_{i=1}^{K} f\left(\mathbf{x}_{i}\right) \tag{5.5}
\end{equation*}
$$

holds for all polynomials $f$ of degree $\leq t$. Hardin and Sloane (1996) provide a set of three dimensional t -designs to order $p=10$. For an exploration of the properties of t -designs and a discussion of methods for creating higher-order designs, see Sloan and Womersley (2008).

With this distribution of points, the acceleration at the sink position, $\mathbf{r}$, can be computed from Gauss' law as

$$
\begin{equation*}
\Psi(\mathbf{r})=\sum_{j=1}^{K} \frac{M_{j}}{\left|\mathbf{r}-\mathbf{R}_{j}\right|^{3}} \hat{\mathbf{r}} \tag{5.6}
\end{equation*}
$$

Makino (1999) and Kawai et al. (2004) exploited this pseudo-particle scheme to develop a high-order Barnes-Hut tree code suited to the GRAPE processor, in which each sink particle interacts with a multiplicity of clusters of pseudo-particles as well as its near field.

Mathematically, the solution of Laplace's equation $\nabla^{2} \phi=0$ within a sphere in three dimensions is uniquely determined by Dirichlet boundary conditions on the surface of the sphere; i.e. $\left.\phi\right|_{S^{2}}=\Phi(\mathbf{R})$. Poisson's formula for a function in three dimensions which is harmonic in the interior and on the boundary of the unit sphere has the following solution,

$$
\begin{equation*}
\phi(\mathbf{r})=\frac{1}{4 \pi} \int_{S^{2}} \Phi(\mathbf{R}) \frac{1-r^{2}}{\left(1+r^{2}-2 r \cos \gamma\right)^{3 / 2}} d S \tag{5.7}
\end{equation*}
$$

where $\mathbf{r}=(r, \theta, \phi)$ with $r<1, \mathbf{R}$ is on the surface of the unit sphere $S^{2}$, i.e. $\mathbf{R}=(1, \hat{\theta}, \hat{\phi})$, and $\gamma$ is the angle between $\mathbf{R}$ and $\mathbf{r}$. If we know $\Phi\left(\mathbf{R}_{i}\right)$ at the t -design locations on the surface of $S^{2}$, then by the definition of a t-design, $\Phi(\mathbf{R})$ can be represented as a polynomial of order $t$ at any point on the surface.

In the limit of an infinite number of particles, the spherical harmonic multipole,

$$
\begin{equation*}
\alpha_{l}^{m}=\frac{4 \pi}{2 l+1} \sum_{i=1}^{N} m_{i} r_{i}^{l} Y_{l}^{m *}\left(\theta_{i}, \phi_{i}\right) \tag{5.8}
\end{equation*}
$$

becomes

$$
\begin{equation*}
\alpha_{l}^{m}=\frac{4 \pi a^{l+2}}{2 l+1} \int_{S^{2}} \rho(a, \theta, \phi) Y_{l}^{m *}(\theta, \phi) d s \tag{5.9}
\end{equation*}
$$

where $\rho$ is the continuous pseudo-particle density. Writing a spherical harmonic expansion as

$$
\begin{equation*}
f(\theta, \phi)=\sum_{n=0}^{\infty} \sum_{m=-n}^{n} a_{m n} Y_{n}^{m}(\theta, \phi) \tag{5.10}
\end{equation*}
$$

with coefficients

$$
\begin{equation*}
a_{m n}=\int_{0}^{2 \pi} \int_{0}^{\pi} d \phi d \theta \sin (\theta) f(\theta, \phi) Y_{n}^{m *}(\theta, \phi) \tag{5.11}
\end{equation*}
$$

one can immediately see that, after truncating the infinite sum to finite order $p$, the pseudoparticle density is

$$
\begin{equation*}
\rho(a, \theta, \phi)=\sum_{l=0}^{p} \sum_{m=-l}^{l} \frac{2 l+1}{4 \pi a^{l+2}} \alpha_{l}^{m} Y_{l}^{m}(\theta, \phi) \tag{5.12}
\end{equation*}
$$

Using the fact that the spherical $t$-design is a set of $K$ points such that

$$
\begin{align*}
\int_{S^{2}} f(\mathbf{s}) d \mathbf{s} & =\int_{0}^{2 \pi} \int_{0}^{\pi} d \phi d \theta \sin (\theta) f(\theta, \phi) \\
& =\frac{4 \pi}{K} \sum_{j=1}^{K} f\left(\theta_{j}, \phi_{j}\right)  \tag{5.13}\\
& =\frac{4 \pi}{K} \sum_{j=1}^{K} f\left(\mathbf{R}_{j}\right)
\end{align*}
$$

we have that the total mass is

$$
\begin{align*}
M & =\frac{4 \pi}{K} \sum_{j=1}^{K} \rho\left(a, \theta_{j}, \phi_{j}\right) a^{2} \\
& =\frac{4 \pi}{K} \sum_{j=1}^{K} \sum_{l=0}^{p} \sum_{m=-l}^{l} \frac{2 l+1}{4 \pi} a^{l+1} \alpha_{l}^{m} Y_{l}^{m}\left(\theta_{j}, \phi_{j}\right) \tag{5.14}
\end{align*}
$$

and since $M=\sum_{j=1}^{K} M_{j}$ one immediately has that the $K$ pseudo-particle masses are given by

$$
\begin{equation*}
M_{j}=\frac{4 \pi}{K} \sum_{l=0}^{p} \sum_{m=-l}^{l} \frac{2 l+1}{4 \pi} a^{l+1} \alpha_{l}^{m} Y_{l}^{m}\left(\theta_{j}, \phi_{j}\right) \tag{5.15}
\end{equation*}
$$

The quantity pre-multiplying $\alpha_{l}^{m}$,

$$
\begin{equation*}
P_{l m, j}=\sum_{l=0}^{p} \sum_{m=-l}^{l} \frac{2 l+1}{4 \pi} a^{l+1} Y_{l}^{m}\left(\theta_{j}, \phi_{j}\right) \tag{5.16}
\end{equation*}
$$

can be precomputed once and for all and the masses can now be found with a matrix multiplication, $M=P \alpha$; this is particularly efficient on the GPU. The only object that remains is the efficient computation of $\alpha_{l}^{m}$ for a cell.

The identity

$$
\begin{equation*}
r^{l} Y_{l}^{m}(\theta, \phi)=\sqrt{\frac{(2 l+1)(l+m)!(l-m)!}{4 \pi}} \sum_{h, j, k=0}^{l} \frac{\delta_{h+j+k, l} \delta_{h-j, m}}{h!j!k!2^{h+j}}(-x-i y)^{h}(x-i y)^{j} z^{k} \tag{5.17}
\end{equation*}
$$

where $x, y, z$ are derived from the relations $\theta=\operatorname{accros}(z / r), \phi=\arctan (x, y)$ and $r^{2}=x^{2}+$ $y^{2}+z^{2}$, demonstrates that that $r^{l} Y_{l}^{m}(\theta, \phi)$ is merely the sum of polynomials of the form $x^{h} y^{j} z^{k}$. Using the Mathematica code in Appendix B, we obtain the coefficients $c_{j k h}$ which satisfy

$$
\begin{equation*}
\alpha_{l}^{m}=\frac{4 \pi}{2 l+1} \sum_{q} m_{q} r_{q}^{l} Y_{l}^{m}\left(\theta_{q}, \phi_{q}\right)=\frac{4 \pi}{2 l+1} \sum_{j+h+k==p} c_{j k h}\left[\sum_{q} m_{q} x_{q}^{j} y_{q}^{h} z_{q}^{k}\right] \tag{5.18}
\end{equation*}
$$

The quantity in brackets is, of course, the Cartesian multipole, and now $\alpha_{l}^{m}$ can be represented as a matrix multiplication, $\alpha=C \times M$, where $C$ is the matrix of coefficients $c_{j k h}$ and $M$ are the Cartesian multipoles. Both the Cartesian multipoles and the matrix multiplication to form $\alpha$ are naturally very quickly computed on the GPU. Thus in practice, despite the compact representation of Makino's expression for the $M_{j}$,

$$
\begin{equation*}
M_{j}=\sum_{i=1}^{N} m_{i} \sum_{l=0}^{p} \frac{2 l+1}{K}\left(\frac{r_{i}}{a}\right)^{l} P_{l}\left(\cos \gamma_{i j}\right) \tag{5.19}
\end{equation*}
$$

(where $\gamma_{i j}$ is the angle between $\mathbf{r}_{i}$ and $\mathbf{R}_{j}$, and $P_{l}$ is the $l$ th Legendre polynomial), our expression (5.15) is at least an order of magnitude faster to compute.

We now turn to determining a set of $K$ masses $M_{i}$, on the surface of a sphere of radius $b$, at positions $\mathbf{R}_{i}=\left(b, \hat{\theta}_{i}, \hat{\phi_{i}}\right)$, such that the induced potential on the surface of sphere of radius $a$, where $a<b$, at positions $\mathbf{S}_{i}=\left(a, \hat{\theta}_{i}, \hat{\phi}_{i}\right)$,

$$
\begin{equation*}
\hat{\phi}\left(\mathbf{S}_{i}\right)=\sum_{j} \frac{M_{i}}{\left|\mathbf{R}_{j}-\mathbf{S}_{i}\right|} \tag{5.20}
\end{equation*}
$$

coincides with a given potential $\phi\left(\mathbf{S}_{i}\right)$.

Recalling Poisson's formula, (5.7), and by differentiating the expression for the generating function of $P_{n}(\mu)$ for $|r|<1$

$$
\begin{equation*}
\frac{1}{\sqrt{1-2 \mu r+r^{2}}}=\sum_{n=0}^{\infty} P_{n}(\mu) r^{n} \tag{5.21}
\end{equation*}
$$

with respect to $r$, multiplying the result by $2 r$, and adding this expression to the equation just shown, one obtains, for $|r|<1$,

$$
\begin{equation*}
\frac{1-r^{2}}{\left(1+r^{2}-2 r \cos \theta\right)^{3 / 2}}=\sum_{n=0}^{\infty}(2 n+1) r^{n} P_{n}(\cos \theta) \tag{5.22}
\end{equation*}
$$

Therefore, Poisson's formula, (5.7), can now be written in the form,

$$
\begin{align*}
\phi(\mathbf{r}) & =\frac{1}{4 \pi} \int_{S} \Phi(\mathbf{R})\left(\sum_{n=0}^{\infty}(2 n+1) r^{n} P_{n}(\cos \gamma)\right) d S  \tag{5.23}\\
& =\frac{1}{4 \pi} \int_{S} \Phi(\mathbf{R})\left(\sum_{n=0}^{\infty}(2 n+1) r^{n} P_{n}\left(\frac{\mathbf{r} \cdot \mathbf{R}}{r}\right)\right) d S
\end{align*}
$$

Using the definition of the t-design permits us to rewrite the expression for $u(\mathbf{r})$, above as,

$$
\begin{equation*}
\phi(\mathbf{r})=\frac{1}{K} \sum_{j=1}^{K} \Phi\left(\mathbf{R}_{j}\right)\left(\sum_{n=0}^{\infty}(2 n+1) r^{n} P_{n}\left(\frac{\mathbf{r} \cdot \mathbf{R}_{j}}{r}\right)\right) \tag{5.24}
\end{equation*}
$$

where $\mathbf{R}_{j}$ are the positions of the $t$-design on the unit sphere $S^{2}$.
Now, with the assumption that the boundary conditions are given at the surface of a sphere of radius $a$, the above equation becomes, for $r<a$,

$$
\begin{equation*}
\phi(\mathbf{r})=\frac{1}{K} \sum_{j=1}^{K} \Phi\left(a \mathbf{R}_{j}\right)\left(\sum_{n=0}^{\infty}(2 n+1)\left(\frac{r}{a}\right)^{n} P_{n}\left(\frac{\mathbf{r} \cdot \mathbf{R}_{j}}{r}\right)\right) \tag{5.25}
\end{equation*}
$$

Using the addition theorem for spherical harmonics

$$
\begin{equation*}
P_{n}(\cos \gamma)=\frac{4 \pi}{2 n+1} \sum_{m=-n}^{n} Y_{n}^{m}(\theta, \phi) Y_{n}^{* m}(\hat{\theta}, \hat{\phi}) \tag{5.26}
\end{equation*}
$$

where $\gamma$ is the angle between two vectors with spherical angles $(\theta, \phi)$ and $(\hat{\theta}, \hat{\phi})$, one obtains, for $r<a$

$$
\begin{equation*}
\phi(\mathbf{r})=\frac{4 \pi}{K} \sum_{j=1}^{K} \Phi\left(a \mathbf{R}_{j}\right)\left(\sum_{n=0}^{\infty}\left(\frac{r}{a}\right)^{n} \sum_{m=-n}^{n} Y_{n}^{m}(\theta, \phi) Y_{n}^{* m}\left(\hat{\theta_{j}}, \hat{\phi_{j}}\right)\right) \tag{5.27}
\end{equation*}
$$

or equivalently,

$$
\begin{equation*}
\phi(\mathbf{r})=\sum_{n=0}^{\infty} \sum_{m=-n}^{n} \alpha_{n}^{m} r^{n} Y_{n}^{m}(\theta, \phi) \tag{5.28}
\end{equation*}
$$

where

$$
\begin{equation*}
\alpha_{n}^{m}=\frac{4 \pi}{K} \sum_{j=1}^{K} \Phi\left(a \mathbf{R}_{j}\right) a^{-n} Y_{n}^{* m}\left(\hat{\theta_{j}}, \hat{\phi_{j}}\right) \tag{5.29}
\end{equation*}
$$

The $\alpha_{n}^{m}$ are the multipole representations of the potential $\Phi\left(a \mathbf{R}_{j}\right)$.
We now need an expression for the multipoles of the pseudo-particles, $\beta_{n}^{m}$. By considering two vectors $\mathbf{r}$ and $\mathbf{R}$ with $|\mathbf{r}|<|\mathbf{R}|$, one then has

$$
\begin{equation*}
\frac{1}{|\mathbf{r}-\mathbf{R}|}=\frac{1}{\sqrt{r^{2}-2 r R \cos \gamma+R^{2}}}=\frac{1}{R} \sum_{n=0}^{\infty} P_{n}(\cos \gamma)\left(\frac{r}{R}\right)^{n} \tag{5.30}
\end{equation*}
$$

where $\gamma$ is the angle between $\mathbf{r}$ and $\mathbf{R}$. Using the Addition Theorem for Spherical Harmonics, for $r<R$,

$$
\begin{equation*}
\frac{1}{|\mathbf{r}-\mathbf{R}|}=\frac{4 \pi}{R} \sum_{n=0}^{\infty} \frac{1}{2 n+1}\left(\frac{r}{R}\right)^{n} \sum_{m=-n}^{n} Y_{n}^{m}(\theta, \phi) Y_{n}^{* m}(\hat{\theta}, \hat{\phi}) \tag{5.31}
\end{equation*}
$$

Therefore,

$$
\begin{align*}
\sum_{i=1}^{K} \frac{M_{i}}{\left|\mathbf{r}-b \mathbf{R}_{i}\right|} & =\sum_{i=1}^{K} M_{i} \frac{4 \pi}{b} \sum_{n=0}^{\infty} \frac{1}{2 n+1}\left(\frac{r}{b}\right)^{n} \sum_{m=-n}^{n} Y_{n}^{m}(\theta, \phi) Y_{n}^{* m}\left(\hat{\theta}_{i}, \hat{\phi}_{i}\right) \\
& =\sum_{n=0}^{\infty} \sum_{m=-n}^{n} r^{n} \beta_{n}^{m} Y_{n}^{m}(\theta, \phi) \tag{5.32}
\end{align*}
$$

where

$$
\begin{equation*}
\beta_{n}^{m}=\frac{4 \pi}{2 n+1} \sum_{i=1}^{K} M_{i} b^{-n-1} Y_{n}^{* m}\left(\hat{\theta}_{i}, \hat{\phi}_{i}\right) \tag{5.33}
\end{equation*}
$$

In order to satisfy 5.20 we require $\alpha_{n}^{m}=\beta_{n}^{m}$, or equivalently,

$$
\begin{equation*}
\alpha_{n}^{m}=\frac{4 \pi}{2 n+1} b^{-n-1} \sum_{i=1}^{K} M_{i} Y_{n}^{m *}\left(\hat{\theta}_{i}, \hat{\phi}_{i}\right) \tag{5.34}
\end{equation*}
$$

As before, if we consider the limit of infinite $K$, and introduce $\rho$, the continuous mass representation for the pseudo-particles (the pseudo-particle density), then we can write

$$
\begin{equation*}
\alpha_{n}^{m}=\frac{4 \pi}{2 n+1} b^{-n-1} \int_{S^{2}} \rho(b, \theta, \phi) b^{2} Y_{n}^{m *}(\theta, \phi) d s \tag{5.35}
\end{equation*}
$$

We can immediately invert this equation; after truncating the infinite sum to finite order $p$, we have the density

$$
\begin{equation*}
\rho(b, \theta, \phi)=\sum_{l=0}^{p} \sum_{m=-l}^{l} \frac{2 l+1}{4 \pi} b^{l-1} \alpha_{l}^{m} Y_{l}^{m}(\theta, \phi) \tag{5.36}
\end{equation*}
$$

By definition, the total mass $M$ is the integral of the surface density, $\rho(b, \theta, \phi)$, times the surface area of a the sphere, $b^{2}$. In other words,

$$
\begin{equation*}
M=\int_{0}^{2 \pi} \int_{0}^{\pi} d \phi d \theta \sin (\theta) \rho(b, \theta, \phi) b^{2} \tag{5.37}
\end{equation*}
$$

Using the definition of a spherical $t$-design, (5.13),

$$
\begin{align*}
M & =\frac{4 \pi}{K} \sum_{j=1}^{K} \rho\left(b, \hat{\theta_{j}}, \hat{\phi_{j}}\right) b^{2} \\
& =\frac{4 \pi}{K} \sum_{j=1}^{K} \sum_{l=0}^{p} \sum_{m=-l}^{l} \frac{2 l+1}{4 \pi} b^{l+1} \alpha_{l}^{m} Y_{l}^{m}\left(\hat{\theta_{j}}, \hat{\phi_{j}}\right) \tag{5.38}
\end{align*}
$$

and since

$$
\begin{equation*}
M=\sum_{j=1}^{K} M_{j} \tag{5.39}
\end{equation*}
$$

one has the pseudo-particle masses

$$
\begin{equation*}
M_{j}=\frac{4 \pi}{K} \sum_{l=0}^{p} \sum_{m=-l}^{l} \frac{2 l+1}{4 \pi} b^{l+1} \alpha_{l}^{m} Y_{l}^{m}\left(\hat{\theta_{j}}, \hat{\phi_{j}}\right) \tag{5.40}
\end{equation*}
$$

Using the relation

$$
\begin{equation*}
\alpha_{l}^{m}=\frac{4 \pi}{K} \sum_{i=1}^{K} \Phi\left(a \mathbf{R}_{i}\right) a^{-l} Y_{l}^{m *}\left(\hat{\theta}_{i}, \hat{\phi}_{i}\right) \tag{5.41}
\end{equation*}
$$

one has

$$
\begin{equation*}
M_{j}=\frac{4 \pi}{K} \sum_{l=0}^{p} \sum_{m=-l}^{l} \frac{2 l+1}{4 \pi} b^{l+1}\left[\frac{4 \pi}{K} \sum_{i=1}^{K} \Phi\left(a \mathbf{R}_{i}\right) a^{-l} Y_{l}^{m *}\left(\hat{\theta_{i}}, \hat{\phi}_{i}\right)\right] Y_{l}^{m}\left(\hat{\theta_{j}}, \hat{\phi}_{j}\right) \tag{5.42}
\end{equation*}
$$

Since

$$
\begin{equation*}
P_{l}\left(\cos \gamma_{i j}\right)=\frac{4 \pi}{2 l+1} \sum_{m=-l}^{l} Y_{l}^{m}\left(\hat{\theta_{j}}, \hat{\phi_{j}}\right) Y_{l}^{m *}\left(\hat{\theta_{i}}, \hat{\phi_{i}}\right) \tag{5.43}
\end{equation*}
$$

we finally have

$$
\begin{equation*}
M_{j}=\sum_{i=1}^{K} \Phi\left(a \mathbf{R}_{i}\right) \sum_{l=0}^{p} \frac{(2 l+1)^{2}}{K^{2}} \frac{b^{l+1}}{a^{l}} P_{l}\left(\cos \gamma_{i j}\right) \tag{5.44}
\end{equation*}
$$

Once again, we can pre-compute the matrix $Q_{i j}$ for a specific t-design

$$
\begin{equation*}
Q_{i j}=\sum_{l=0}^{p} \frac{(2 l+1)^{2}}{K^{2}} \frac{b^{l+1}}{a^{l}} P_{l}\left(\cos \gamma_{i j}\right) \tag{5.45}
\end{equation*}
$$

once and for all and simply find $M_{i}=Q_{i j} \Phi_{j}$ by matrix multiplication on the GPU. This is the crucial and last step of our formulation of a pseudo-particle FMM scheme.

An important aspect of the formulation is that the error distribution for a particular order is typically much better than the corresponding order in an FMM expansion. The reasons for this amount to the fact that the number of pseudo-particles for a given order is typically overestimated by a significant fraction - therefore viewed in terms of a polynomial expansion we are effectively working at a higher order. The error distributions are shown in Figure (5.5). As a typical example, for an order $p=8$ expansion, the best attempt at a t-design is $K=144$ particles, and we can evaluate on the order of 3.22 million cell-cell pseudo-particle interactions per second on an NVIDIA GTX 295, and 1.16 million per second for $p=10$ with $K=240$. This rate is directly proportional to $K^{-2}$ with the constant of proportionality being rate of direct potential evaluations - about 60 billion per second.


Figure 5.5: The two top panels illustrate the maximum relative errors in a pseudo-particle expansion for a single source particle at $(0.5,0.5,0.5)$ and a uniform distribution of 1024 source particles at varying separations of $|\vec{R}|$ and orders $p=2,4,6,8,10$ (from the top of the graph) as computed by Monte Carlo sampling. The bottom two panels show the distribution of relative errors in the acceleration for the pseudo-particle expansion at a separation of $\vec{R}=2$ and order $p=16$ for the single source particle in two $y-z$ planes, $x=1.5$ and $x=2.5$ respectively.

In conclusion, we have constructed a key element of our total programme, namely to use create a tree-code method which can obtain machine precision accelerations and use the GPU effectively. When the degree of clustering in our cosmological samples is high we will need to use this formulation; this is precisely what we will now discuss in the next chapter.

## CHAPTER 6

## ABACUS: An N-body code for Computational Cosmology

Having introduced in previous chapters the mathematics of both infinite lattice sums and higher order multipole expansions in the context of the Fast Multipole Method (FMM), in this chapter we discuss the implementation of our code, ABACUS. We begin by partitioning the simulation volume into into a $K \times K \times K$ cubic lattice, where $K>1$ is an odd integer. Denoting each of the cells by a triple $(i, j, k)$ where $i, j, k=0 . . K-1$, the centers of these cells is denoted by $\mathbf{c}_{i j k}$, and the set of particles within these cells by $\left\{G_{i j k}\right\}$, we induce a partitioning of the cells into those which are, and are not, well-separated from the center cell. This induces the far and near fields, respectively (see Figure 6.1). $K$ is chosen so as to balance as best as possible the computational effort devoted to the near and far fields.

Without loss of generality, the following derivation is concerned only with computing the acceleration in the central cell of the simulation volume; periodicity, and the assumption that $K$ is odd, allows any cell to be rotated to the center. Naturally, following such a rotation, the infinite replicas of this lattice then rearrange themselves in an identical fashion. Under the assumption that the simulation volume is unity and centered on the origin, to obtain the far field potential at position $\mathbf{r}$ in the center cell, one can write the contribution from all cells in periodic replicas and from the well-separated cells within the simulation volume as

$$
\begin{align*}
\phi^{F a r}(\mathbf{r}) & =\sum_{i j k} \sum_{\mathbf{n}}^{\prime} \sum_{l \in\left\{G_{i j k}\right\}} \frac{m_{l}}{\left|\mathbf{r}_{l}+\mathbf{n}-\mathbf{r}\right|}  \tag{6.1}\\
& =\sum_{i j k} \sum_{\mathbf{n}}^{\prime} \sum_{l \in\left\{G_{i j k}\right\}} \frac{m_{l}}{\left|\left(\mathbf{r}_{l}-\mathbf{c}_{i j k}\right)-\left(\mathbf{r}-\mathbf{c}_{i j k}-\mathbf{n}\right)\right|}
\end{align*}
$$

The primed summation symbol denotes that cells in the simulation volume, i.e. with


Figure 6.1: The arrangement of cells contributing to the far field potential $\phi^{F a r}$. This is a slice through the center of the cubical simulation volume. The large square at the center is the simulation volume itself; the other large squares are its periodic replicas. The smaller squares are the partition of the simulation volume, here a $7 \times 7 \times 7$ lattice, and their own periodic replicas. The cells in grey contribute to the far-field acceleration. The cross-hatched cells in the simulation volume are well-separated from the center cell (solid black), as are all cells in the periodic replicas. $\phi^{F a r}$ thus contains the contribution from all but the ring of white squares, the 26 nearest neighbors of the central cell. Our method allows us to compute $\phi^{F a r}$ to very high accuracy with no contribution from the central 27 cells.
$\mathbf{n}=(0,0,0)$, are excluded if they are not well-separated from the center cell; i.e. the sum includes all cells in gray in Figure (4.1). The summation on $i j k$ is over all cells in the partition, and that on $l$ is over the particles in each set $\left\{G_{i j k}\right\}$. Thus $\phi^{F a r}$ accounts for the contribution from particles in all cells which are well-separated from the center cell in the simulation volume, and in all cells of all periodic images of the volume.

Using the expansion (4.9), we have

$$
\begin{equation*}
\phi^{F a r}(\mathbf{r})=\sum_{i j k} \sum_{\substack{\alpha \beta \gamma \\ p=0}}^{\infty} M_{i j k}^{\alpha \beta \gamma}\left(\mathbf{c}_{i j k}\right) \sum_{\mathbf{n}}^{\prime} \mathcal{D}^{\alpha \beta \gamma}\left(\mathbf{c}_{i j k}-\mathbf{n}-\mathbf{r}\right) \tag{6.2}
\end{equation*}
$$

where the $\mathcal{M}_{i j k}^{\alpha \beta \gamma}$ are the reduced multipole moments of cell $i j k$ with respect to its center $\mathbf{c}_{i j k}$ :

$$
\begin{equation*}
M_{i j k}^{\alpha \beta \gamma}=\sum_{l \in\left\{G_{i j k}\right\}} m_{l} \mathcal{M}^{\alpha \beta \gamma}\left(\mathbf{r}_{l}-\mathbf{c}_{i j k}\right) \tag{6.3}
\end{equation*}
$$

The multipoles in any cell are the same for all replicas of that cell; hence one can factor the $M_{i j k}^{\alpha \beta \gamma}$ out of the sum over replicas in 6.2. Expanding the expression for the derivative in a Taylor series for $|\mathbf{r}|<\left|\mathbf{r}^{\prime}\right|$

$$
\begin{equation*}
\mathcal{D}^{\alpha \beta \gamma}\left(\mathbf{r}^{\prime}-\mathbf{r}\right)=\sum_{\delta \varepsilon \zeta} \frac{-1^{\delta+\varepsilon+\zeta}}{\delta!\varepsilon!\zeta!} r_{x}^{\delta} r_{y}^{\varepsilon} r_{z}^{\zeta} \mathcal{D}^{(\alpha+\delta)(\beta+\varepsilon)(\gamma+\zeta)}\left(\mathbf{r}^{\prime}\right) \tag{6.4}
\end{equation*}
$$

we now have

$$
\begin{equation*}
\phi^{F a r}(\mathbf{r})=\sum_{i j k} \sum_{\substack{\alpha \beta \gamma \\ p=0}}^{\infty} M_{i j k}^{\alpha \beta \gamma} \sum_{\substack{\delta \varepsilon \zeta \\ p=0}}^{\infty} \frac{-1^{\delta+\varepsilon+\zeta}}{\delta!\varepsilon!\zeta!} r_{x}^{\delta} r_{y}^{\varepsilon} r_{z}^{\zeta} \Theta_{i j k}^{(\alpha+\delta)(\beta+\varepsilon)(\gamma+\zeta)} \tag{6.5}
\end{equation*}
$$

where

$$
\begin{equation*}
\Theta_{i j k}^{\alpha \beta \gamma}=\sum_{\mathbf{n}}^{\prime} \mathcal{D}^{\alpha \beta \gamma}\left(\mathbf{c}_{i j k}+\mathbf{n}\right) \tag{6.6}
\end{equation*}
$$

This is an important simplification; the quantity $\Theta_{i j k}^{\alpha \beta \gamma}$ is now the only one which contains a sum over the infinite lattice of replicas. Since these derivatives depend only on the size of the partition $K$, they can be computed once and for all for each $K$. For any position $\mathbf{r}$ in any cell $i j k$, the expression

$$
\begin{align*}
\Psi^{F a r}(\mathbf{r})= & \sum_{\substack{\alpha \beta \gamma \\
p=0}}^{\infty}\left(r_{x}-c_{i j k, x}\right)^{\alpha}\left(r_{y}-c_{i j k, y}\right)^{\beta}\left(r_{z}-c_{i j k, z}\right)^{\gamma} \\
& \times\left(\begin{array}{c}
(\alpha+1) T_{i j k}^{(\alpha+1), \beta, \gamma} \\
(\beta+1) T_{i j k}^{\alpha,(\beta+1), \gamma} \\
(\gamma+1) T_{i j k}^{\alpha, \beta,(\gamma+1)}
\end{array}\right) \tag{6.7}
\end{align*}
$$

is the contribution to the acceleration from all particles in cells which are well-separated from the cell containing $\mathbf{r}$ and in all of the cells in the simulation volume's periodic images. The infinite lattice sum has been transformed into a Taylor series whose coefficients are a cyclic convolution of the multipole moments with the summed derivatives $\Theta$. The Taylor coefficients are expressed as

$$
\begin{equation*}
T_{i j k}^{\alpha \beta \gamma}=\frac{(-1)^{(\alpha+\beta+\gamma)}}{\alpha!\beta!\gamma!} \sum_{\substack{\delta \varepsilon \zeta \\ p=0}}^{\infty} \sum_{i^{\prime} j^{\prime} k^{\prime}} M_{i^{\prime} j^{\prime} k^{\prime} k^{\prime}}^{\delta \varepsilon \zeta} \Theta_{\left[i-i^{\prime}\right] K}^{(\alpha+\delta)(\beta+\varepsilon)\left(\gamma+j^{\prime}\right]_{K}\left[k-k^{\prime}\right]_{K}} \tag{6.8}
\end{equation*}
$$

where the symbols $\left[i-i^{\prime}\right]_{K}$ denote modulo $K$ and embody the wrapping-around of the cell indices under periodic boundary conditions. These Taylor coefficients, comprise a cyclic convolution in space (over $i j k$ ), and a linear convolution in multipole index (over $\alpha \beta \gamma$ ). Denoting by $\mathbf{x}$ and $\mathbf{y}$ two arbitrary vectors of $n$ components, the linear convolution of $\mathbf{x}$ and $\mathbf{y}$, denoted by $\mathbf{w}=\mathbf{x} \star \mathbf{y}$, is a vector of $2 n-1$ components such that, for $0 \leq i \leq 2 n-1$,

$$
\begin{equation*}
w_{i}=\sum_{j=\max \{0, i-n+1\}}^{\min \{i, n-1\}} x_{j} y_{i-j} \tag{6.9}
\end{equation*}
$$

where the upper and lower bounds in the summation are chosen in such a way that the indices $j$ and $i-j$ always range between 0 and $n-1$. The cyclic convolution denoted by $\mathbf{x} \otimes \mathbf{y}$ as a vector $\mathbf{z}$ of $n$ components, such that for $0 \leq i \leq n-1$,

$$
\begin{equation*}
z_{i}=\sum_{j=0}^{n-1} x_{j} y_{(i-j) \bmod n} \tag{6.10}
\end{equation*}
$$

The conventional scheme for evaluating a linear convolution employs the result that

$$
\begin{equation*}
\mathbf{w}=\mathrm{FFT}_{2 n}^{-1}\left(\mathrm{FFT}_{2 n}\left(\mathbf{x} \mid \mathbf{0}_{n}\right) \odot \mathrm{FFT}_{2 n}\left(\theta \mid \mathbf{0}_{n}\right)\right) \tag{6.11}
\end{equation*}
$$

where $\odot$ denotes component-wise product, and $\left(\mathbf{x} \mid \mathbf{0}_{k}\right)$ denotes the vector obtained by padding vector $\mathbf{x}$ with $k$ zeros. The number of pairwise products of reduced multipoles and derivatives for an order $P$ expansion is $(1+P)(2+P)(3+P)(4+P)(5+2 P) / 120 \propto P^{5} ;$
for orders $4,8,12$, and 16 , this is $182,2079,10556$, and 35853 , respectively. Since $P$ is of modest size, typically $P \leq 16$, we evaluate the linear convolution as a direct sum.

A cyclic convolution is evaluated via

$$
\begin{equation*}
\mathbf{z}=\mathrm{FFT}_{n}^{-1}\left(\mathrm{FFT}_{n}(\mathbf{x}) \odot \mathrm{FFT}_{n}(\mathbf{y})\right) \tag{6.12}
\end{equation*}
$$

Therefore, simply computing the FFT of $\mathbf{x}$ and $\mathbf{y}$ with no padding, multiplying their components, and then taking the inverse FFT gives us the cyclic convolution of $\mathbf{x}$ and $\mathbf{y}$.

The spatial cyclic convolution may thus be computed as

$$
\begin{equation*}
\hat{T}^{\alpha \beta \gamma}=\sum_{\substack{\delta \varepsilon \zeta \\ p=0}}^{\infty} \hat{M}^{\delta \varepsilon \zeta} \odot\left(\hat{\Theta}^{(\alpha+\delta)(\beta+\varepsilon)(\gamma+\zeta)}\right)^{*} \tag{6.13}
\end{equation*}
$$

where $\hat{X}$ denotes the Fourier transform of the quantity $X$, and $X^{*}$ its complex conjugate. The symbol $\odot$ denotes element-wise multiplication of the vectors of length $K^{2}(K+1) / 2$ (recall that the inherent symmetry in the Fourier transform of a real sequence can be exploited so that one is only required to evaluate the pairwise complex multiplications for approximately half the elements). The cyclic convolution implicitly computes the Taylor coefficients in each cell as if the cell were rotated to the center, applying the periodic boundary condition and wrapping coördinates correctly back into the simulation volume. This transforms an otherwise $O\left(K^{6}\right)$ operation into one of $O\left(K^{3} \log K\right)$. One need only form the inverse Fourier transform of the $(P+1)^{2}$ independent quantities $\hat{T}^{\alpha \beta \gamma}$, for $\gamma=$ 0,1 ; once again, the remaining $T^{\alpha \beta \gamma}$ may be obtained via the trace-free recursion relation.

The implicit rotation of cells to the center of the simulation volume introduces a slight complication to the Redlack-Grindlay term, $\zeta_{R G}(\mathbf{r})$; as each cell is rotated to the center, the dipole moment of the simulation volume changes. Let $\mathbf{r}$ be the position of any particle within the simulation volume. Under the coördinate rotation which brings the cell center $\mathbf{c}_{i j k}$ to the origin, denote the new position of $\mathbf{r}$ as $W_{i j k}(\mathbf{r})$. Assume that $\mathbf{r}^{\prime}$ is a particle
position in cell $C_{i j k}$ and that we have rotated this cell to the center. The new coördinates of $\mathbf{r}^{\prime}$ are $\mathbf{r}=W_{i j k}\left(\mathbf{r}^{\prime}\right)$, and we have

$$
\begin{align*}
\zeta_{i j k}^{R G}(\mathbf{r}) & =\frac{4 \pi}{3}\left(\sum_{l=1}^{N} m_{l} W_{i j k}\left(\mathbf{r}_{l}\right)-\mathbf{r} M\right) \\
& =\frac{4 \pi}{3}\left(\sum_{a b c} \sum_{l \in\left\{G_{a b c}\right\}} m_{i} W_{i j k}\left(\mathbf{r}_{l}\right)-\mathbf{r} M\right) \tag{6.14}
\end{align*}
$$

Using the result that for any particle with position $\mathbf{r}$, in cell $a b c, W_{i j k}(\mathbf{r})=\mathbf{r}+W_{i j k}\left(\mathbf{c}_{i j k}-\right.$ $\mathbf{c}_{a b c}$ ),

$$
\begin{align*}
\zeta_{i j k}^{R G}(\mathbf{r}) & =\frac{4 \pi}{3}\left(\sum_{a b c} \sum_{l \in\left\{G_{a b c}\right\}} m_{l}\left(\mathbf{r}_{l}+W_{i j k}\left(\mathbf{c}_{i j k}-\mathbf{c}_{a b c}\right)\right)-\mathbf{r} M\right)  \tag{6.15}\\
& =\frac{4 \pi}{3}\left(\sum_{l} m_{l} \mathbf{r}_{l}+\sum_{a b c} M_{a b c} W_{i j k}\left(\mathbf{c}_{i j k}-\mathbf{c}_{a b c}\right)-\mathbf{r} M\right)
\end{align*}
$$

where $M_{i j k}$ is the mass in cell $(i, j, k)$ and $M$ is the total mass of the simulation volume. The first and third terms above are straightforward to compute. The only term which varies as cells are rotated to the center is the second term. This term can be written as

$$
\begin{equation*}
\sum_{a b c} M_{a b c} W_{i j k}\left(\mathbf{c}_{i j k}-\mathbf{c}_{a b c}\right)=\sum_{a b c} M_{a b c} \mathbf{c}_{[i-a]_{K}[j-b]_{K}[k-c]_{K}} \tag{6.16}
\end{equation*}
$$

from which it is clearly seen that this term is again a circular convolution and can be evaluated using the Fourier convolution theorem.

In the previous section we have presented the mathematics of the far field computation. A naive implementation would dictate that one requires $3(p+1)^{2} K^{2}(K+1) / 2$ elements for the three arrays $\hat{\Theta}, \hat{M}, \hat{T}$ at an order $p$ expansion of the far field. Instead, we demonstrate that only approximately $2 K^{2}(K+1)$ elements are required for the computation to proceed, irrespective of $p$.

We first show that for any fixed order on an $L^{3}$ grid, one only needs space for $(p+$ $1)^{2} L^{2}(L+1) / 2$ elements plus a small amount of temporary storage which can be made
arbitrarily small. Recall that the Fourier transformed quantity, $\hat{T}$, is a sum of pairwise products in Fourier space of the multipoles, $\hat{M}$, and derivatives, $\hat{\Theta}$, over all cells. We can construct $\hat{T}$ for any subset of the cells, $B$, which we will denote a block, by

$$
\begin{equation*}
\forall x, y, z \in B \quad \hat{T}^{\alpha \beta \gamma}(B)=\sum_{\substack{\delta \varepsilon \zeta \\ p=0}}^{\infty} \hat{M}^{\delta \varepsilon \zeta}(B) \odot\left(\hat{\Theta}^{(\alpha+\delta)(\beta+\varepsilon)(\gamma+\zeta)}\right)^{*}(B) \tag{6.17}
\end{equation*}
$$

This expression reveals that once the $\hat{T}(B)$ for block $B$ are accumulated, the $\hat{M}(B)$ are no longer required for any other block $\hat{T}\left(B^{\prime}\right)$, and hence we can store the $\hat{T}$ in the space previously occupied by $\hat{M}$; this is illustrated in figure (6.2).

This method requires temporary space to expand the reduced representation into the complete set (only quantities with indices where $\alpha+\beta+\gamma=p \wedge \gamma \in 0,1$ are stored) using the recurrence relation satisfied by $\hat{M}$ and $\hat{\Theta}$. This requires $\|B\|(p+1)(p+2)(p+3) / 6$ elements (where $\|B\|$ denotes the cardinality of $B$ ) for each of the expanded $\hat{M}$ and $\hat{\Theta}$ arrays, denoted by $\hat{M}_{\text {temp }}$ and $\hat{\Theta}_{\text {temp }}$ respectively, and $\|B\|(p+1)^{2}$ elements for $\hat{T}$ temporary storage. The multipoles, $M$, and their Fourier transform, $\hat{M}$ are therefore only computed once. The Fourier transform of the derivatives, $\hat{\Theta}$, must be computed for each reduced order as usual, however we are now discarding $L^{2}(L+1) / 2-B$ elements and storing the remaining quantities in $\hat{\Theta}_{\text {temp }}$; thus in this scheme the Fourier transform of the derivatives must be computed $L^{2}(L+1) /(2 B)$ times. This cost of these $L^{2}(L+1) /(2 B)$ FFT's is negligible compared to cost of the convolution provided that $B$ is not too small. In fact, the block size $B$ should be chosen so that the temporary arrays $\hat{M}_{\text {temp }}, \hat{\Theta}_{\text {temp }}, \hat{T}_{\text {temp }}$ all fit into cache on the processor (12 MB for the Q9550).

There is an additional computational device to aid us in computing the far field for large $K$ - a multipole hierarchy. Consider our problem of convolving on a $K^{3}$ grid with an order 16 expansion, denoted by $C_{K^{3}, 16}$. Assume that this does not quite fit in memory, even using the factor of three savings in space of the computational device just de-


Figure 6.2: Memory layout for the convolution. We have schematically illustrated a blocking factor, $\|B\|=\left(L^{2}(L+1) / 2\right) / 4$. Each of the vertical lines represents all $L^{2}(L+1) / 2$ cells at a given multipole order. The $\hat{M}_{\text {temp }}$ and $\hat{\Theta}_{\text {temp }}$ require more multipole orders as they have been expanded into their Cartesian representations. $\hat{M}$ originally contains the Fourier transform of the reduced multipoles. As we accumulate each Taylor series for a block of size $\|B\|$ in $\hat{T}_{\text {temp }}$ we overwrite the corresponding multipoles in $\hat{M}$. After 4 such iterations the entire $\hat{T}$ has replaced $\hat{M}$.
scribed. Assume further, however, that $C_{K^{3}, 8}$ does fit into memory, and that we perform this convolution. One can now perform $F^{3}$ convolutions, where $F \geq 2, C_{L^{3}, 16}$, and where $L=K / F+2 \times 4$. The extra 4 cells on each side of the $[K / 2]^{3}$ sub-volume is the kernel width surrounding a cell that was not included during the order 8 convolution.

Naturally this is a recursive procedure; if $C_{K^{3}, 8}$ had not fit we would have tried $C_{K^{3}, 4}$ and performed a 3-level recursive hierarchy. This scheme is extremely effective in exploiting the idea that for any cell, outside a certain radius, $r_{p}$ (the kernel width), an order $p$ expansion can be replaced with an order $p^{\prime}$ expansion, where $p^{\prime}<p$. Typically for
orders $16,8,4,2$ we choose kernel widths $r_{p}$ of $2,4,16,64$ respectively. The only penalty which we incur is a loss of efficiency in comparison to directly evaluating $C_{K^{3}, p}$. The efficiency is asymptotically $\left(L /\left(L+2 \times R\left(p^{\prime}\right)\right)\right)^{3}$, where $p^{\prime}$ is the order employed in the level immediately preceding level $p$ in the hierarchy.

Finally, we consider the extension of this algorithm to a distributed memory computing cluster: the "Beowulf" architecture. This is arguably the most common form of commodity supercomputing today. The essential criterion for parallel scalability is locality of communication between processors. In the current context, one would wish to avoid broadcasting all of the multipole information on every processor to all other processors. A naive implementation of the algorithm described above would do just that, as the convolution would require using a distributed FFT. Massively parallel cosmological simulations commonly use distributed FFT's, despite their notoriously high communication overhead. (As discussed previously, many of the codes used are based upon a PM-style algorithm.)

Assume we have $P^{3}$ processors (cores) where $P$ is odd and we desire the Taylor coefficients in each cell due to all other cells (except for near neighbors) on a $G^{3}$ distributed grid where $G=F P$ with G odd ( we can generalize this algorithm to a $P \times Q \times R$ processors going to a $F_{1} P \times F_{2} Q \times F_{3} R$ grid, however the details are tedious and would obscure an understanding of the underlying algorithm we present below.) For ease of illustration let us assume we are working at order 16 with a kernel width of 1 around ourselves. By performing a $P \times P \times P$ convolution as described in this chapter reduces the processor's remaining interactions to those of it's 27 nearest processors (including itself). (We assume that each processor has computed the multipoles for all cells in it's entire volume (on an $F^{3}$ grid) and propagated this information up the hierarchy of cells until we reach the single cell representing the entire volume).

To obtain the Taylor coefficients for the $F^{3}$ grid on our current processor we could perform a $27 F^{3}$ open convolution (recall that only the initial convolution was required to take into account periodicity). The open convolution incurs an additional factor of 64/9 for any grid size in comparison to the periodic problem. Additionally, we would need to receive $26 F^{3}(p+1)^{2}$ numbers; the multipoles in all the cells in our neighbors. One can do significantly better with a hierarchical method, which we illustrate (all the partitions are drawn in cross section) in Figure (6.3) and now describe. We begin by considering an open convolution of size $9^{3}$ where we form the Taylor series for all 27 cells in our volume by assuming all multipoles in the $3^{3}$ cells in our partition and those immediately adjacent to our volume are zero (so as not to double count their contribution upon repartitioning). We then partition each these cells where we have assumed the multipoles are zero by a factor of 27; again assuming all multipoles in our volume and those immediately adjacent to us are zero. This process is repeated until the target $F^{3}$ grid is reached. This will require $F / 3$ applications of this algorithm. In the figure we have partitioned our volume 4 times so that there are $81^{3}$ cells in our volume. Assuming each machine holds on the order of 1 billion particles we need one more partitioning (however this is too fine a grid to illustrate in the figure).

In Figure (6.4) we show the rate of computation on an Intel Q9950 processor for the far field convolution; starting with already-computed multipole moments and summed derivatives, this is the total time to produce the set of Taylor coefficients which describe the far field throughout the computational volume.

The only aspect of the discussion which remains concerns the choice of $K$, which is completely determined by the degree of clustering of the particular distribution. One measure of the degree of clustering at $z=0$ is to consider the ratio of the cube root of the number of particles to the size of the simulation volume's linear dimension. We quantify
the degree of clustering by considering the number of particles per cell in our $K^{3}$ lattice.
In Figure 6.5 we illustrate the cell population distribution for $K=32,64,128$ at $z=0$ of two cosmological test problems from Heitmann et al. (2005), $\Lambda C D M$ simulations using $256^{3}$ particles in a $256 \mathrm{Mpc} / \mathrm{h}$ box and $64 \mathrm{Mpc} / \mathrm{h}$ respectively, with $\Omega_{\text {tot }}=0.314, \Omega_{\Lambda}=$ 0.686 , and $H_{0}=71 \mathrm{~km} \mathrm{~s}^{-1} \mathrm{Mpc}^{-1}$. Consider the statistic, $W_{K}$, which is a measure of the total number of direct interactions,

$$
\begin{equation*}
W_{K}=\sum_{a=0}^{K-1} \sum_{b=0}^{K-1} \sum_{c=0}^{K-1} \sum_{l \in C_{a, b, c}} \sum_{i=-1}^{1} \sum_{j=-1}^{1} \sum_{k=-1}^{1} \sum_{j \in C_{[i+a]_{K},[j+b]_{K},[k+c]_{K}}} m_{j} \frac{\mathbf{r}_{j}-\mathbf{r}_{l}}{\left|\mathbf{r}_{j}-\mathbf{r}_{l}\right|^{3}} \tag{6.18}
\end{equation*}
$$

where the symbols $[a+b]_{K}$ denote modulo $K$ arithmetic and embody the notion of wrappingaround of the cell indices under periodic boundary conditions, and $C_{i j k}$ refers to the cell with index $i j k, 0 \leq i, j, k<K$ within the simulation volume. The statistics for the 256 $\mathrm{Mpc} / \mathrm{h}$ box are $W_{32}=3.15 \times 10^{11}, W_{64}=6.428 \times 10^{10}, W_{128}=1.99 \times 10^{10}$, and for the 64 $\mathrm{Mpc} / \mathrm{h}$ box $W_{32}=8.84 \times 10^{11}, W_{64}=5.03 \times 10^{11}, W_{128}=2.66 \times 10^{11}$. Theoretically (one can typically achieve only a fraction of peak speed of a GPU; say 70\%) the statistic $W_{K}$ translates into a computational rate for obtaining accelerations of $4.5 \times 10^{10} / W_{K}$ particles per second for a single GTX 295 GPU. Naturally, as $W_{K}$ is a monotonically decreasing function of $K$, it's dual $F_{K}$ (which is the computational cost of obtaining the far field on a grid of size $K$ ) is a monotonically increasing function of $K$. Whereas one would consider the case of $256^{3}$ particles in a $256 M p c / h$ box (a similar degree of clustering to the Horizon problem) to be modest clustering and amenable to direct interactions for the near field, the $64 M p c / h$ simulation exhibits a higher degree of clustering and therefore $\left(W_{K}\right)$ would become prohibitively expensive as the mass resolution increases thus necessitating the use of the FMM tree scheme of the previous chapter for the near field.

The culmination of this formalism can be seen in the results of the cosmological pancake collapse (Zel'Dovich, 1970; Shandarin and Zeldovich, 1989). This is of particular
interest because it is a sensitive test of accuracy, and especially as Heitmann et al. (2005) use this problem to compare several current cosmological simulation codes. The parameters for this problem were $\Omega_{0}=1, H_{0}=50 \mathrm{~km} / \mathrm{s} / \mathrm{Mpc}$, in a $\sqrt{3} \cdot 10 \mathrm{Mpc}$ box with $64^{3}$ particles. The initial conditions were chosen so that the redshift at which the first collapse occurs is $z_{c}=5$, and the pancake normal is aligned with the box diagonal, i.e. inclined at 54.7 degrees with respect to the faces of the simulation volume. The smoothing length chosen was 15 kpc . These results are shown in Figure 2, and were computed using parameters $K=33, P=16, R=1$. This test problem demonstrates that even with modest parameters excellent accuracy can be obtained in comparison with any of the codes used by Heitmann et al. (2005).

We next computed one of the cosmological test problems from Heitmann et al. (2005), a $\Lambda C D M$ simulation using $256^{3}$ particles in a $256 \mathrm{Mpc} / \mathrm{h}$ box with $\Omega_{t o t}=0.314, \Omega_{\Lambda}=$ 0.686, and $H_{0}=71 \mathrm{~km} \mathrm{~s}^{-1} \mathrm{Mpc}^{-1}$. The parameters for our algorithm were $P=12$, $R=1$ and $K=63$. The co-moving smoothing length was 64 kpc . At $z=50$, the computation proceeded at approximately $10^{6}$ particles/second. By $z=1$, the rate was $8 \times 10^{5}$ particles/second, and at $z=0$ the rate was $6 \times 10^{5}$ particles/second. The formation of clustering is the culprit for the decrease in rate of computation as the simulation proceeds, as the number of direct interactions required for the near field clearly increases with clustering. This computation was performed on a single Intel Q9950 (quad core) processor with 8GB of RAM. The output was compared with that from the code $M C^{2}$, and the agreement is excellent. A slice through both simulations is shown in Figures (6.7) and (6.8).

The code has been used to perform numerous run of $1024^{3}$ particles in a $(1024 M p c)^{3}$ box on a computer with 128 GB of memory and four quad-core Intel Xeon E5540 processors. The typical computational rate was $2 \times 10^{6}$ particles per second down to $z=3$; by $z=0$, the rate had decreased to $5 \times 10^{5}$ particles per second due to the onset of clustering.

These simulations are now being used to study systematic effects in baryon acoustic oscillations surveys of large-scale structure using JDEM. For a single timestep at $z=0$ the time to perform the force evaluation using a GTX 280 was approximate 250 seconds, for a rate of about 4 million particles per second. This rate is to be compared to an inferred rate of computation of the Horizon simulation (Teyssier et al., 2008) ( $4096^{3}$ particles in $2 \mathrm{Gpc} / \mathrm{h}$ box) of approximately 25,000 particles per second per node. We derived this quantity by assuming that the forces for all 70 billion particles were calculated at the finest time step (the authors stated that there were effectively approximately 10,000 time steps at the finest level) on 6144 processors for 2 months. Whilst we acknowledge that there is inefficiency due to the massive parallelism in this calculation we still have a factor of approximately 200 increase in performance.

An important aspect of our method is dramatically reduced memory requirements; whilst the Millennium-II simulation reported requiring 8TB for $2170^{3}$ particles ( 10 billion particles; effectively 800 bytes per particle), a conservative estimate for our adaptively gridded scheme is 100 bytes per particle or 1 TB . 1 TB of memory can be acquired today with 16 machines each with 64 GB of memory for a total cost of $\$ 60,000$. The situation for large scale structure simulations such as Kim et al. (2008) (70 billion particles in a 6.5 Gpc box) is even more amenable to commodity computation; we only require 44 bytes per particle (as no tree information is required), and current hardware pricing for 16 machines each with 256 GB should cost a total of $\$ 300,000$. The direct implication of using a small number of machines is that we should be able to avoid the difficulties associated with load balancing that PKDGRAV and GADGET must entertain.


Figure 6.3: Parallel algorithm for the far field as described in the text.


Figure 6.4: Computational Rate of the far field convolution as a function of $K$ and order on a single Intel Q9950 processor. From the top of the graph the lines represent orders $16,12,8$, and 4 respectively.


Figure 6.5: Cell populations as a function of the number of cells used to grid the simulation volume from left to right $32^{3}, 64^{3}, 128^{3}$ cells; the two simulations are $z=0$ for Heitmann $256^{3}$ particles in 360 Mpc and 90 Mpc respectively.


Figure 6.6: Zeldovich pancake collapse test problem computed with $N=64^{3}$ particles. The black points are all of the particles in the simulation. The red line is a onedimensional particle-mesh solution computed with $2^{17}$ points. (cf. Heitmann et al. (2005) figure 3)


Figure 6.7: A slice from the $z=0$ simulation of the Heitmann et al. (2005) $256^{3}$ particle in $256 \mathrm{Mpc} h^{-1}$ LCDM test problem, computed from the code described in the text.


Figure 6.8: The same slice from the output from the MC2 code as presented in Heitmann et al. (2005).

## CHAPTER 7

## Conclusions and Discussion

Large sets of high-accuracy, high-resolution cosmological simulations are a critical component of the "precision cosmology" program as well as to the study of the evolution of galaxies, clusters, and the intergalactic medium. While N -body simulations do not include baryonic physics, using a statistical model of the relation between galaxy properties and their dark matter halos, environments, and merger histories (Ma and Fry, 2000; Seljak, 2000; Peacock and Smith, 2000; Scoccimarro et al., 2001; Cooray and Sheth, 2002; Bullock et al., 2002; van den Bosch et al., 2003; Berlind et al., 2003, and many others), one can use these simulations to make large catalogs of mock galaxies. In addition, N body simulations increasingly are the basis for semi-analytic models of galaxy formation (Kauffmann et al., 1993; Cole et al., 1994; Somerville and Primack, 1999; Hatton et al., 2003; Baugh, 2006; Croton et al., 2006; De Lucia et al., 2006).

From an experimental point of view, the use of large N -body simulations to produce mock catalogs is an essential component in the experimental design of cosmological surveys. As surveys become larger and more sensitive, simulations must keep up, modeling larger volumes of the universe and providing more detailed predictions of smaller structures. Simulations are also a critical element of the observational program, as they provide the connection between fundamental model parameters and statistical measures from observations, as well as the ability to characterize systematic biases and estimate errors and covariances for data analysis. Large suites of high-accuracy, high-resolution cosmological simulations are a critical component of the "precision cosmology" program, rather than one single heroic run to quantify cosmic variance and to establish dependencies on cosmological parameters. One typically needs dozens of runs to study parameter de-
pendencies (e.g. Heitmann et al., 2009) and hundreds to thousands of runs to establish high-quality covariance matrices (e.g. Takahashi et al., 2009).

Most current cosmological simulation codes have abandoned attempts at computing the periodic acceleration directly from the particle positions and instead employ methods which are variants of the traditional Particle Mesh (PM) scheme. This choice has been based, in large measure, on the assumption that PM schemes are faster than the equivalent tree representations.

In this dissertation we have computed the periodic acceleration as a sum of direct interactions. We can compute an Ewald-equivalent acceleration to arbitrary accuracy far more rapidly than any other scheme. The computational efficiency of the method comes from the fact that the $|\mathbf{r}|^{-1}$ kernel is very much simpler than the kernel for direct interactions employed by codes like PKDGRAV or GADGET-2. The main virtue of our direct interaction representation is that N -body problems with periodic boundary conditions can be trivially reformulated as an equivalent problem with open (or isolated) boundary conditions.

Recognizing that these direct interactions can be implemented very efficiently on the GPU, we introduced an analytic representation of the FMM using only direct interactions. In addition, we implemented a fast algorithm for the GPU to obtain adaptive softening lengths for collisionless N-body simulations. Beyond pure N-body simulations, the ability to obtain softening lengths, and therefore smoothing lengths for SPH, together with the fact that SPH is based purely upon kernel evaluations between two particles, has permitted a particularly straightforward implementation of SPH on the GPU. With the techniques implemented in this dissertation, in the near future we would like to construct a coupled SPH N-body with cooling which completely runs on the GPU, in particular to test predictions of the detectability of the effects of baryonic cooling on dark matter halo
profiles (which lead to percent level changes in the convergence angular power spectrum at wavenumbers of several thousand) in future weak lensing surveys.

Finally, whereas state-of-the-art "heroic" simulations such as the Via-Lactea and Aquarius simulations (as implemented via PKDGRAV-2 and Gadget-3 respectively) require supercomputer centers with 1000's of processors for the equivalent of several million CPU hours, GPUs computing $|\mathbf{r}|^{-1}$ kernel evaluations are three orders of magnitude less expensive for equivalent rates of force evaluations per second. It is our opinion that in the future one should never need massively parallel systems such as Blue Gene or even massive supercomputers such as Roadrunner to perform cosmological simulations. The smaller-scale commodity hardware we advocate (GPUs) is not only faster but constitutes only a modest investment at the scale of a department or university.

## Appendix A

## Cartesian-To-Reduced Multipole Transformation

```
(* The potential function }\phi(\mathbf{r})*
Phi[\mp@subsup{x}{-}{\prime},\mp@subsup{y}{-}{\prime},\mp@subsup{z}{-}{\prime}]:= 1/Sqrt[x^2+\mp@subsup{y}{}{\wedge}2+\mp@subsup{z}{}{\wedge}2]
(* Make all (a,b,c) with }a+b+c=k,c\in(0,1) *)
ReducedTuples[k_] :=
    Select[Partition[Flatten[Table[{a,b,c},{a,0,k},{b,0,k},{c,0,1}]],
        3],(#[[1]]+#[[2]] +#[[3]] == k) &]
(* Make all tuples (a,b,c) with }a+b+c=k *
CartesianTuples[k_] :=
    Select[Partition[Flatten[Table[{a,b,c},{a,0,k},{b,0,k},{c,0,k}]],
        3],( #[[1]]+#[[2]]+#[[3]]\[Equal] k)&]
(* Make reduced multipoles M}\mp@subsup{M}{}{abc}\mathrm{ by repeated differentiation of }\phi(\mathbf{r})\quad*
ReducedMultipole[a_,b_, c_] :=
    Expand[Factor[
        Sqrt[x^2+y^2+\mp@subsup{z}{}{\wedge}2\mp@subsup{]}{}{\wedge}(2*(a+b+c)+1)* Derivative[a,b,c][Phi][x,y,z]]]
(* Solve for M}\mp@subsup{M}{}{a,b,c}\mathrm{ as linear combinations of Cartesian multipoles Q Qbc *)
CartesianToReduced[n_] := Module[
    {1,R,C,eq,vars },
    R = ReducedTuples[n];
    C = CartesianTuples[n];
    eq = R/. {a_,b_,c_} -> ReducedMultipole [a,b,c];
    vars = C/ . {a_,b-, c_ } -> x^a*y^b*z^c;
    l = Outer[Coefficient,eq,vars]
]
KC[ n_] := (( }\textrm{n}+1)*(\textrm{n}+2))/2(* number of Q Q with a+b+c=n *)
KR[\mp@subsup{n}{-}{}]:=(2*n+1) (* number of M Mac}\mathrm{ with }a+b+c=n,c\in(0,1) *
(* print table of coefficients *)
CF[\mp@subsup{n}{-}{\prime}]:= Module[
    {1,v},
    l = CartesianToReduced [n];
    v = Table[Subscript[a,i],{i,0,KC[n]-1}];
    CForm[1.v]
]
(* write code to evaluate the linear combinations with these coefficients *)
CreateCartesianToReducedCode[n_] :=
    Module[
        {strm},
            strm = OpenWrite["Cartesian2Reduced.c"];
        WriteString[strm, "void Reduced0(double *a, double *r) {r[0] = a[0]; } \n"];
```

WriteString[strm,"void Reduced1 (double *a, double *r)
$\{\backslash \mathrm{n} ", " \mathrm{r}[0]=-\mathrm{a}[0] ; \mathrm{r}[1]=-\mathrm{a}[1] ; \mathrm{r}[2]=-\mathrm{a}[2] ;\} \backslash \mathrm{n} "] ;$
Do[WriteString[strm,"void Reduced", 1 ," (double *a, double *r)

$$
\{\backslash \mathrm{n} ", " \quad \text { int } \mathrm{i} ; \backslash \mathrm{n} ", " \text { double } \mathrm{v} ", 1, "[]=\{\backslash \mathrm{n} "] ;
$$

WriteString[strm,
StringReplace [StringReplace [ ToString [CF[1]],"List ("->" "],"))"->")"] ];

"[i];","\n\}\n\n"],\{1,2,n\}];
Close[strm]

## ApPENDIX B

## Cartesian-To-Spherical Harmonic Transformation

Mathematica code to produce C code to perform the Cartesian-to-SphericalHarmonic multipole transformation:

```
CartesianTuples[k_] :=
    Select[Partition[Flatten[Table[{a, b, c}, {a, 0, k}, {b, 0, k}, {c, 0, k}]],
        3], ( #[[1]] + #[[2]] + #[[3]] == k) &]
KC[\mp@subsup{n}{-}{}]:= ((n + 1)*(n + 2))/2
KY[n_] := (2*n + 1)
RealPart[\mp@subsup{z}{-}{\prime}]:= ComplexExpand[Re[z]]
ComplexPart[\mp@subsup{z_]}{_}{\prime}:= ComplexExpand[\boldsymbol{Im}[z]]
CartesianSphericalHarmonicY[{1_, m_}, {x-, y-, z_}] :=
    Factor[Together[TrigExpand[ExpToTrig[
```



```
            Sqrt[1 - z^2/(x^2 + y^2 + z^^2)]/Sqrt[1 + y^^/x^2] ->
                Sqrt[(x^2 + y^2)/(x^2 + y^2 + z^2)]/Sqrt[( (x^2 + y^2)/x^2] /
        Sqrt[(x^2 + y^2)/(x^2 + y^2 + z^2)]/Sqrt[(x^2 + y^2)/x^2] - 
            x/Sqrt[x^2 + y^2 + z^2]
RY[n_] := N[
            Table [ComplexExpand
                Expand[CartesianSphericalHarmonicY[{n, m}, {x, y, z}] * (x^2 + y^2 + z^2)^(n/2)]], {m, -n, n}]]
C2RealY[n_] :=
    Module[ {1, R, C, vars }, R = Flatten[RealPart[RY[n]l];
        C = CartesianTuples[n]; eq = R; vars = C /. {a_, b_, c_} :> x^a y^b z^c;
        l = Outer[Coefficient, eq, vars]]
SubC2RealY[n_] :=
    Module[{1, v}, l = C2RealY[n];
        v = Table[Subscript[c, i], {i, 0, KC[n] - 1}]; CForm[l.v]]
OutputCartesian2RealY[n_] :=
    Module[{strm}, strm = OpenWrite["C:\Cartesian2RealY.c"];
        WriteString[strm, "#define Subscript(c,i) c[i] \n"];
        WriteString[strm
            "void RealY0(double *c, double *y) { y[0] = 0.5/sqrt(M_PI); } \n"];
            Do[WriteString[strm, "void RealY", l, "(double *c, double *y) {\n",
                        int i; \n", "double v", l, "[] = {\n"];
            WriteString[strm,
                    StringReplace [StringReplace [ToString[SubC2RealY [ 1]], "List ("-> ""],
```

```
        "))" -> ")"]];
    WriteString[strm, "\n};\n"," for(i=0;i<", KY[l],";i++) y[i] = v"
    1, "[i];", "\n}\n\n"], {1, 1, n}]; Close[strm]]
C2ComplexY[\mp@subsup{n}{-}{\prime}]:=
    Module[ {l, R, C, vars }, R = Flatten[ComplexPart[RY[n]]];
        C=CartesianTuples[n]; eq = R; vars = C /. {a_, b_, c_} :> x^a y^b z^c ;
    l = Outer[Coefficient, eq, vars]]
SubC2ComplexY[n_] :=
    Module[{1, v }, l = C2ComplexY[n];
        v = Table[Subscript[c, i], {i, 0, KC[n] - 1}]; CForm[1.v]]C
OutputCartesian2ComplexY[n_] :=
    Module[{strm}, strm = OpenWrite["C:\Cartesian2ComplexY.c"];
        WriteString[strm, "#define Subscript(c,i) c[i] \n"];
        WriteString[strm ,
            "void ComplexY0(double *c, double *y) { y[0] = 0; } \n"];
        Do[WriteString[strm, "void ComplexY", l, "(double *c, double *y) {\n",
            int i; \n","double v", l,"[] = {\n"];
        WriteString[strm,
            StringReplace [
                StringReplace [ ToString [ SubC2ComplexY [ 1]], "List(" -> ""],
                "))" -> ")"]];
            WriteString[strm, "\n};\n"," for(i=0;i<", KY[l],";i++) y[i] = v",
                1, "[i];", "\n}\n\n"], {1, 1, n}]; Close[strm]]
```


## REFERENCES

Alder, B. J. and T. E. Wainwright (1959). Studies in molecular dynamics I. General method. J. Chem Phys., 31, pp. 459-466.

Anderson, C. R. (1992). An Implementation of the Fast Multipole Method without Multipoles. SIAM Journal on Scientific and Statistical Computing, 13(4), pp. 923-947. doi:10.1137/0913055.

Athanassoula, E., E. Fady, J. C. Lambert, and A. Bosma (2000). Optimal softening for force calculations in collisionless N-body simulations. MNRAS, 314, pp. 475-488.

Bagla, J. S. (2002). TreePM: A Code for Cosmological N-Body Simulations. Journal of Astrophysics and Astronomy, 23, pp. 185-196.

Barnes, J. and P. Hut (1986). A Hierarchical O(NlogN) Force-Calculation Algorithm. Nature, 324, pp. 446-449.

Baugh, C. M. (2006). A primer on hierarchical galaxy formation: the semi-analytical approach. Reports on Progress in Physics, 69, pp. 3101-3156.

Beatson, R. and L. Greengard (1997). A short course on fast multipole methods. In Wavelets, Multilevel Methods and Elliptic PDEs, pp. 1-37. Oxford University Press.

Berlind, A. A., D. H. Weinberg, A. J. Benson, C. M. Baugh, S. Cole, R. Davé, C. S. Frenk, A. Jenkins, N. Katz, and C. G. Lacey (2003). The Halo Occupation Distribution and the Physics of Galaxy Formation. ApJ, 593, pp. 1-25.

Borwein, D., J. M. Borwein, and K. F. Taylor (1985). Convergence of lattice sums and

Madelung's constant. Journal of Mathematical Physics, 26(11), pp. 2999-3009. doi: 10.1063/1.526675.

Bouchet, F. R. and L. Hernquist (1988). Cosmological simulations using the hierarchical tree method. Astrophys. J., Suppl. Ser., 68, pp. 521-538.

Boylan-Kolchin, M., V. Springel, S. D. M. White, A. Jenkins, and G. Lemson (2009). Resolving Cosmic Structure Formation with the Millennium-II Simulation. ArXiveprints.

Bullock, J. S., R. H. Wechsler, and R. S. Somerville (2002). Galaxy halo occupation at high redshift. MNRAS, 329, pp. 246-256.

Chau, N. H., A. Kawai, and T. Ebisuzaki (2008). Acceleration of Fast Multipole Method Using Special-Purpose Computer Grape. Int. J. High Perform. Comput. Appl., 22(2), pp. 194-205.

Cipriani, B., J. \& Silvi (1982). Molecular Physics, 45, p. 259.

Cole, S., A. Aragon-Salamanca, C. S. Frenk, J. F. Navarro, and S. E. Zepf (1994). A Recipe for Galaxy Formation. MNRAS, 271, pp. 781-+.

Cooray, A. and R. Sheth (2002). Halo models of large scale structure. Phys. Rep., 372, pp. 1-129.

Croton, D. J., V. Springel, S. D. M. White, G. De Lucia, C. S. Frenk, L. Gao, A. Jenkins, G. Kauffmann, J. F. Navarro, and N. Yoshida (2006). The many lives of active galactic nuclei: cooling flows, black holes and the luminosities and colours of galaxies. MNRAS, 365, pp. 11-28.

Davé, R., J. Dubinski, and L. Hernquist (1997). Parallel TreeSPH. New Astronomy, 2, pp. 277-297.
de Leeuw, S. W., J. W. Perram, and E. R. Smith (1980). Simulation of Electrostatic Systems in Periodic Boundary Conditions. I. Lattice Sums and Dielectric Constants. Proc. R. Soc. Lond. A, 373, pp. 27-56.

De Lucia, G., V. Springel, S. D. M. White, D. Croton, and G. Kauffmann (2006). The formation history of elliptical galaxies. MNRAS, 366, pp. 499-509.

Dehnen, W. (2001). Towards optimal softening in three-dimensional N-body codes - I. Minimizing the force error. MNRAS, 324, pp. 273-291.

Dehnen, W. (2002). A Hierarchical O(N) Force Calculation Algorithm. Journal of Computational Physics, 179, pp. 27-42.

Dubinski, J. (1996). A parallel tree code. New Astronomy, 1, pp. 133-147.

Dubinski, J., J. Kim, C. Park, and R. Humble (2004). GOTPM: a parallel hybrid particlemesh treecode. New Astronomy, 9, pp. 111-126.

Ewald, P. (1921). Die Berechnung Optischer und Elektrostatische Gitterpotentiale. Ann. Phys., 64, pp. 253-287.

Fincham, D. (1994). Optimisation of the Ewald Sum for Large Systems. Molecular Simulation, 13, pp. 1-9.

Fukushige, T., J. Makino, and A. Kawai (2005). GRAPE-6A: A Single-Card GRAPE-6 for Parallel PC-GRAPE Cluster Systems. PASJ, 57, pp. 1009-1021.

Gibbs, J. W. and E. B. Wilson (1901). Vector Analysis. New Haven:Yale University Press, c 1901 .

Greengard, L. and V. Rokhlin (1987). A Fast Algorithm for Particle Simulations. J. Comput. Phys., 73, p. 325.

Gumerov, N. A. and R. Duraiswami (2008). Fast multipole methods on graphics processors. J. Comput. Phys., 227(18), pp. 8290-8313.

Hardin, R. H. and N. J. Sloane (1996). McLaren's Improved Snub Cube and Other New Spherical Designs in Three Dimensions. Discrete and Computational Geometry, 15, pp. 429-441.

Hatton, S., J. E. G. Devriendt, S. Ninin, F. R. Bouchet, B. Guiderdoni, and D. Vibert (2003). GALICS- I. A hybrid N-body/semi-analytic model of hierarchical galaxy formation. MNRAS, 343, pp. 75-106.

Heitmann, K., D. Higdon, M. White, S. Habib, B. J. Williams, and C. Wagner (2009). The Coyote Universe II: Cosmological Models and Precision Emulation of the Nonlinear Matter Power Spectrum. ArXiv e-prints.

Heitmann, K., P. M. Ricker, M. S. Warren, and S. Habib (2005). Robustness of Cosmological Simulations. I. Large-Scale Structure. "Astrophys. J., Suppl. Ser.", 160, pp. 28-58.

Hernquist, L., F. R. Bouchet, and Y. Suto (1991). Application of the Ewald method to cosmological N-body simulations. Astrophys. J., Suppl. Ser., 75, pp. 231-240.

Hinsen, K. and B. U. Felderhof (1992). Reduced description of electric multipole potential in Cartesian coordinates. J. Math. Phys., 33, pp. 3731-3735.

Kauffmann, G., S. D. M. White, and B. Guiderdoni (1993). The Formation and Evolution of Galaxies Within Merging Dark Matter Haloes. MNRAS, 264, pp. 201-+.

Kawai, A., J. Makino, and T. Ebisuzaki (2004). Performance Analysis of High-Accuracy Tree Code Based on the Pseudoparticle Multipole Method. "Astrophys. J., Suppl. Ser.", 151, pp. 13-33.

Kim, J., C. Park, J. R. I. Gott, and J. Dubinski (2008). The Horizon Run N-body Simulation: Baryon Acoustic Oscillations and Topology of Large Scale Structure of the Universe. ArXiv e-prints.
M. Eleftheriou, M., B. Fitch, A. Rayshubskiy, T. Ward, and R. Germain (2005). Scalable framework for 3D FFTs on the Blue Gene/L supercomputer: Implementation and early performance measurements. IBM J. of Res. and Dev., 49, pp. 457+.

Ma, C.-P. and J. N. Fry (2000). Halo Profiles and the Nonlinear Two- and Three-Point Correlation Functions of Cosmological Mass Density. ApJ, 531, pp. L87-L90.

Makino, J. (1999). Yet Another Fast Multipole Method without MultipolesPseudoparticle Multipole Method. Journal of Computational Physics, 151, pp. 910920.

Merz, H. (2006). Scaling Performance of CubePM and TVD MHD on Blue Gene. World Wide Web electronic publication. Www-03.ibm.com/systems/resources/systems_deepcomputing_pdf_Blue_Gene_Applications_Paper_CubePM_032306.pdf.

Merz, H., U.-L. Pen, and H. Trac (2005). Towards optimal parallel PM N-body codes: PMFAST. New Astronomy, 10, pp. 393-407.

Navarro, J. F., A. Ludlow, V. Springel, J. Wang, M. Vogelsberger, S. D. M. White, A. Jenkins, C. S. Frenk, and A. Helmi (2008). The Diversity and Similarity of Cold Dark Matter Halos. ArXiv e-prints.

Peacock, J. A. and R. E. Smith (2000). Halo occupation numbers and galaxy bias. MNRAS, 318, pp. 1144-1156.

Peebles, P. J. E. (1980). The large-scale structure of the universe. Research supported by the National Science Foundation. Princeton, N.J., Princeton University Press, 1980. 435 p.

Price, D. J. and J. J. Monaghan (2007). An energy-conserving formalism for adaptive gravitational force softening in smoothed particle hydrodynamics and N -body codes. MNRAS, 374, pp. 1347-1358.

Redlack, A. and J. Grindlay (1975). The Coulomb Potential in a Finite Slab: Orthorhombic Symmetry. J. Phys. Chem. Solids, 36, pp. 953-965.

Rybicki, G. B. (1986). A Gridless Fourier Method. In Hut, P. and S. L. W. McMillan (eds.) The Use of Supercomputers in Stellar Dynamics, volume 267 of Lecture Notes in Physics, Berlin Springer Verlag, p. 181.

Salmon, J. K. and M. S. Warren (1994). Skeletons from the Treecode Closet. J. Comp. Phys., 111, pp. 136-155.

Sangster, M. J. L. and M. Dixon (1976). Interionic potentials in alkali halides and their use in simulations of the molten salts. Adv. Phys., 25, pp. 247-342.

Scoccimarro, R., R. K. Sheth, L. Hui, and B. Jain (2001). How Many Galaxies Fit in a

Halo? Constraints on Galaxy Formation Efficiency from Spatial Clustering. ApJ, 546, pp. 20-34.

Seljak, U. (2000). Analytic model for galaxy and dark matter clustering. MNRAS, 318, pp. 203-213.

Shandarin, S. F. and Y. B. Zeldovich (1989). The large-scale structure of the universe: Turbulence, intermittency, structures in a self-gravitating medium. Reviews of Modern Physics, 61, pp. 185-220.

Shirokov, A. and E. Bertschinger (2005). GRACOS: Scalable and Load Balanced P3M Cosmological N-body Code. preprint,astro-ph/0505087.

Sirko, E. (2005). Initial Conditions to Cosmological N-Body Simulations, or, How to Run an Ensemble of Simulations. ApJ, 634, pp. 728-743.

Sloan, I. H. and R. Womersley (2008). A variational characterisation of spherical designs. UNSW Applied Mathematics Report AMR 08/7, p. 1.

Smith, E. (1981). Proceedings Royal Society London A, 375, p. 476.

Somerville, R. S. and J. R. Primack (1999). Semi-analytic modelling of galaxy formation: the local Universe. MNRAS, 310, pp. 1087-1110.

Springel, V. (2005). The cosmological simulation code GADGET-2. Mon. Not. R. Astron. Soc., 364, pp. 1105-1134.

Springel, V., J. Wang, M. Vogelsberger, A. Ludlow, A. Jenkins, A. Helmi, J. F. Navarro, C. S. Frenk, and S. D. M. White (2008). The Aquarius Project: the subhaloes of galactic haloes. MNRAS, 391, pp. 1685-1711.

Springel, V., N. Yoshida, and S. D. M. White (2001). GADGET: a code for collisionless and gasdynamical cosmological simulations. New Astronomy, 6, pp. 79-117.

Stadel, J. G. (2001). Cosmological N-body simulations and their analysis. Ph.D. Thesis, University of Washington.

Sugimoto, D., Y. Chikada, J. Makino, T. Ito, T. Ebisuzaki, and M. Umemura (1990). A Special-Purpose Computer for Gravitational Many-Body Problems. Nature, 345, pp. 33-+.

Takahashi, R., N. Yoshida, M. Takada, T. Matsubara, N. Sugiyama, I. Kayo, A. J. Nishizawa, T. Nishimichi, S. Saito, and A. Taruya (2009). Simulations of Baryon Acoustic Oscillations II: Covariance matrix of the matter power spectrum. ArXiveprints.

Teyssier, R., S. Pires, S. Prunet, D. Aubert, C. Pichon, A. Amara, K. Benabed, S. Colombi, A. Refregier, and J.-L. Starck (2008). Full-Sky Weak Lensing Simulation with 70 Billion Particles. ArXiv e-prints.
van den Bosch, F. C., X. Yang, and H. J. Mo (2003). Linking early- and late-type galaxies to their dark matter haloes. MNRAS, 340, pp. 771-792.

Ying, L., G. Biros, and D. Zorin (2004). A kernel-independent adaptive fast multipole algorithm in two and three dimensions. J. Comp. Phys., 196, pp. 591-626.

Zel'Dovich, Y. B. (1970). Gravitational instability: An approximate theory for large density perturbations. A\&A, 5, pp. 84-89.

Zemp, M., J. Diemand, M. Kuhlen, P. Madau, B. Moore, D. Potter, J. Stadel, and L. Widrow (2009). The graininess of dark matter haloes. MNRAS, 394, pp. 641-659.


[^0]:    Dissertation Director: Philip A. Pinto

[^1]:    ${ }^{1}$ Since the effect of the cosmological terms solved by N -body simulations is merely to modify the scale factor on distances, we shall henceforth, without loss of generality, examine the canonical N-body problem in the context of periodic boundaries.

[^2]:    ${ }^{1}$ The quad-double package by Hida, Li , and Bailey is available from http://crd.lbl.gov/~dhbailey/mpdist/

