# TREECODES FOR POTENTIAL AND FORCE APPROXIMATIONS

A Dissertation

by

KASTHURI SRINIVASAN KANNAN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

August 2008

Major Subject: Computer Science

# TREECODES FOR POTENTIAL AND FORCE APPROXIMATIONS

A Dissertation

by

KASTHURI SRINIVASAN KANNAN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

| | |
|---|---|
| Co-Chairs of Committee, | Vivek Sarin<br>Devanayagam Palaniappan |
| Committee Members, | Nancy Amato<br>Paul Nelson |
| Head of Department, | Valerie Taylor |

August 2008

Major Subject: Computer Science

# ABSTRACT

Treecodes for Potential and Force Approximations. (August 2008)

Kasthuri Srinivasan Kannan, B.Sc., University of Madras;
M.Sc., Indian Institute of Technology, Madras;
M.S., Texas A&M University

Co–Chairs of Advisory Committee: Dr. Vivek Sarin
Dr. Devanayagam Palaniappan

$N$-body problems encompass a variety of fields such as electrostatics, molecular biology and astrophysics. If there are $N$ particles in the system, the brute force algorithm for these problems based on particle-particle interaction takes $O(N^2)$, which is clearly expensive for large values of $N$. There have been some approximation algorithms like the Barnes-Hut Method and the Fast Multipole Method (FMM) proposed for these problems to reduce the complexity. However, the applicability of these algorithms are limited to operators with analytic multipole expansions or restricted to simulations involving low accuracy. The shortcoming of $N$-body treecodes are more evident for particles in motion where the movement of the particles are not considered when evaluating the potential. If the displacement of the particles are small, then updating the multipole coefficients for all the nodes in the tree may not be required for computing the potential to a reasonable accuracy. This study focuses on some of the limitations of the existing approximation schemes and presents new algorithms that can be used for $N$-body simulations to efficiently compute potentials and forces. In the case of electrostatics, existing algorithms use Cartesian coordinates to evaluate the potentials of the form $r^{-\lambda}$, where $\lambda \geq 1$. The use of such coordinates to separate the variables results in cumbersome expressions and does not exploit the inherent spherical symmetry found in these kernels. For such potentials, we provide a new multipole expansion series and construct a method which is asymptotically superior than the current treecodes. The advantage of this expansion series is further demonstrated by an algorithm that can compute the forces to the desired accuracy. For particles in motion, we introduce a new method in which we retain the multipole coefficients when performing multipole updates (to the parent nodes) at every time step. This results in considerable savings in time while maintaining the accuracy. We further illustrate the efficiency of our algorithms through numerical experiments.

To my father

# ACKNOWLEDGMENTS

There are several people who have been of immense help during the years of my graduate study at Texas A&M. This dissertation would be insignificant without their mention. First, I would like to thank Dr. Vivek Sarin, my graduate advisor, for his support and guidance all these years. His integral view on research and "only high-quality work and not less" attitude, has made a deep impression on me. I owe him lots of gratitude for having shown me this way of research. Besides being an excellent researcher, he is a nice person to work with and I am really glad to have had him as my advisor. My special thanks would go to Dr. Nancy Amato, for being my mentor during the early years of my graduate life. I would have never completed my doctoral studies if not for her support. I have always held a special status for her and if I can make another dedication for this thesis, I would like to dedicate it to Dr. Amato. I thank Dr. Paul Nelson for his commitment in being my committee in spite of having completed his services to the Computer Science department. He was always approachable and was willing to see the progress of my studies. I am indebted to Dr. Palaniappan who monitored my work and took efforts in discussing several ideas related to my research. In addition to the good discussions that we had, the camaraderie we had developed will be cherished for years to come.

My colleagues in the department, Thomas, Meiqiu, Hemant and Radhika, all gave me the feeling of being at home at work - thank you guys. Especially I would like to thank Thomas for his moral support and patience in hearing all my lamentations!

I am extremely grateful to my wife Lavanya for her love and patience during the PhD. period. She made every effort for me to concentrate on my work. I am indebted to my mother for sharing my vision. She had been a great source of inspiration for my study all these years. Also, I would like to thank my sisters and in-laws for their constant encouragement.

I wish to thank the Computer Science department and the secretaries for assisting me in many different ways. I would also like to thank Knowledge Based Systems, Inc. for providing me an opportunity to work as an intern during my student life. This has helped me grow in several ways. I spent some wonderful time doing research at Texas A&M at Qatar and I thank Dr. Jim Holste for giving me that opportunity. Finally, I feel a deep sense of gratitude for my late father to whom I dedicate this dissertation.

# TABLE OF CONTENTS

CHAPTER Page

I INTRODUCTION: $N$-BODY SIMULATIONS . . . . . . . . . . . 1

- A. Introduction to $N$-body Simulations . . . . . . . . . . . . . 1
  - 1. $N$-body Simulation Methods . . . . . . . . . . . . . . . 3
  - 2. Overview of the Treecodes . . . . . . . . . . . . . . . . . 7
  - 3. The Barnes-Hut Method . . . . . . . . . . . . . . . . . . 9
- B. Some Limitations of Treecodes . . . . . . . . . . . . . . . . . 11
- C. Scope of this Dissertation . . . . . . . . . . . . . . . . . . . 14

II A TREECODE FOR POTENTIALS OF THE FORM $r^{-\lambda}$* . . . 15

- A. Multipole Expansion Theorem for $r^{-1}$ Potential . . . . . . 16
- B. Ultraspherical Polynomials . . . . . . . . . . . . . . . . . . . 18
- C. A Treecode for $r^{-\lambda}$ Potentials . . . . . . . . . . . . . . . . 21
- D. Analysis of the Algorithm . . . . . . . . . . . . . . . . . . . 22
  - 1. Complexity . . . . . . . . . . . . . . . . . . . . . . . . . 22
  - 2. Comparisons with Other Methods . . . . . . . . . . . 22
- E. Numerical Experiments . . . . . . . . . . . . . . . . . . . . . 25

III AN ACCURATE FORCE CALCULATION ALGORITHM . . . 32

- A. Introduction to Force Approximations . . . . . . . . . . . . 32
- B. An Expansion Theorem for Forces . . . . . . . . . . . . . . . 34
- C. A Tree Based Scheme for Force Computations . . . . . . . 38
- D. Numerical Experiments . . . . . . . . . . . . . . . . . . . . . 40

IV A FAST TREECODE FOR PARTICLE SIMULATIONS . . . . 44

- A. The First Addition Theorem for Multipole Expansions . . 45
- B. An Expansion Theorem for a Particle in Motion . . . . . . 47
- C. Discussion . . . . . . . . . . . . . . . . . . . . . . . . . . . . 53
- D. A Fast Treecode for Particle Simulations . . . . . . . . . . 54
- E. Numerical Experiments . . . . . . . . . . . . . . . . . . . . . 60

V CONCLUSION AND FUTURE DIRECTIONS . . . . . . . . . . 66

- A. Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . . 66
- B. Future Directions . . . . . . . . . . . . . . . . . . . . . . . . 67

Page

REFERENCES . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 69

APPENDIX A . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 73

VITA . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 77

# LIST OF TABLES

TABLE Page

I Number of particles in the leaf box Vs execution time (in secs) . . . . 27

II Execution time (in secs) for the Coulomb potential . . . . . . . . . . 28

III $RMS$ error (in percentage) for the Coulomb potential . . . . . . . . 28

IV $RMS$ error (in percentage) for the London dispersion potential . . . 29

V $MAC$ Vs Error for 1000 particles of positive masses . . . . . . . . . . 41

VI $MAC$ Vs Error for 10000 particles of positive and negative charges . 42

VII $MAC$ Vs Execution time (in secs) for 10000 particles . . . . . . . . . 42

VIII Error and the time for multipole updates for the treecodes varying $\alpha$ 63

IX Error and the time for multipole updates for treecodes varying $N$ . . 63

X Error and the time for multipole updates for treecodes varying $p$ . . . 64

# LIST OF FIGURES

FIGURE Page

1 Recursive division of the domain and oct-tree representation . . . . . 8

2 Geometry of the distance criteria introduced by Salmon et al. . . . . 11

3 Multipole expansion for a cluster of charge particles . . . . . . . . . 17

4 System size Vs execution time . . . . . . . . . . . . . . . . . . . . . . . 26

5 $\lambda$ Vs $B_\lambda$ . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30

6 Force acting on a point P. Note that $\mathbf{r_{12}} = \mathbf{r} - \rho_{\mathbf{1}}$ . . . . . . . . . . . 38

7 Multipole expansion for forces . . . . . . . . . . . . . . . . . . . . . . . 39

8 Movement of a particle from location $L_1$ to $L_2$ . . . . . . . . . . . . . 48

9 Cell $C$ and its ancestor cell $A$ containing a particle $Q$. . . . . . . . . 58

10 Illustration of multipole updating phase for a particle with $p = 4$. . . 59

# CHAPTER I

# INTRODUCTION: $N$-BODY SIMULATIONS

## A. Introduction to $N$-body Simulations

$N$-body simulations have become very important in theoretical and experimental analysis of complex physical systems. A typical $N$-body problem models physical domains where there is a system of $N$ particles (or bodies), each of which is influenced by gravitational or electrostatic forces due to other particles. Simulation is an essential aspect in areas such as astrophysics or molecular dynamics because the distance and time scales involved makes observations and experiments difficult. It is easy to see that many physical structures can be modelled as a system consisting of $N$ bodies. For example, in astrophysics, the solar system, star clusters and clusters of galaxies can be represented as a collection of bodies in space. Therefore, $N$-body simulation methods can be used to generate theoretical predictions about these systems that can be validated and compared to the observed phenomena. In general, $N$-body simulation methods try to capture the dynamics of the interacting bodies as close as possible. These methods vary depending on the system being studied irrespective of the fundamental similarities. For instance, although the underlying force law in astrophysics (Newton's Law of Universal Gravitation) is governed by same potential function (Coulomb potential) used in the evaluation of long range electrostatic effects in molecular dynamics, the methods employed in these areas are diverse. However, most of the simulations that study the physical systems adhere to the principle of Conservation of Energy as a measure to verify the effectiveness of the simulation. Other physical principles such as Conservation of Angular Momentum are also used

---

The journal model is *Journal of Computational Physics.*

to check the behavior of a $N$-body simulation.

In mathematical terms, a $N$-body problem is an initial value problem consisting of each particle's equation of motion. This can be expressed by a system of differential equations where each particle's initial conditions can be specified by its charge (or mass), its initial position and initial velocity. In such systems, time is considered as an independent variable and the physical quantities like the position, velocity and the acceleration depend on time. Historically, $N$-body problems have been studied by considering the two-body problem, three-body problem and many-body problem (or $N$-body problem). The two-body problem and some special cases of the three body problem have analytic solutions, however there is no general solution for the $N$-body problem. Therefore, the solution of a general $N$-body problem ($N > 2$) is confined to numerical methods of approximation where the system of differential equations which describe particles motion, are discretized into difference equations and the calculation of the trajectories are carried out at regular time intervals using integrators.

Computing force interactions in a $N$-body simulation is not hard in the sense of computational complexity as it follows a second-order polynomial growth. The naive brute force algorithm to compute the forces based on particle-particle interaction is an $O(N^2)$ algorithm. There are various approximation algorithms with lower complexity that have been proposed for these simulations. In the simulation of gravitational forces in astrophysics, Barnes-Hut algorithm [4] and its variants are used. For electrostatic force simulations occurring in molecular dynamics (MD), particle-mesh methods like P3M [10] and Ewald Summation techniques [9] are widely applied. However, the total execution time for a simulation is largely dependent on the size of the system rather than the small difference in the computational complexity of these algorithms. A vast amount of CPU time is required for computing the force interactions even if an $O(N)$ algorithm is used for a modest number of time steps.

Therefore, distributed algorithms have been extensively used in the area of $N$-body simulations.

$N$-body simulations are challenging not only because of the intense computation involved in evaluating the forces during time steps, but also due to the necessity in conserving certain physical quantities like energy and momentum associated with the system. There has been a rapid advance in the area of geometric numerical integration which addresses the later aspect in $N$-body simulations. For instance, several symplectic integrators has been devised in the past decade as they are known for their effectiveness in conserving the phase space geometry of Hamiltonian dynamics. We now have considerable numerical schemes like symplectic Runge-Kutta methods and symplectic Euler methods that have been specifically proposed to preserve the underlying topology of dynamical systems. However, the efficiency of a simulation (both in terms of time and accuracy) is largely determined by its efficient way of computing the interactions as much as it depends on using a conservative integrator. Even in the case of symplectic integrators, it is already known that they are often implicit and require more function evaluations and smaller time steps to produce the same computational accuracy as compared to standard non-symplectic integrators. Therefore, the importance of efficiently computing the interactions in simulations cannot be undermined. In the following section, we will review some of the methods used for computing the interactions in $N$-body simulations.

## 1. $N$-body Simulation Methods

**Particle-Particle (PP):** This is the simplest of all the methods. In general, this method consists of the steps given in Algorithm 1. Although PP method is the most straight-forward method for calculating the forces, there are some drawbacks other than the $O(N^2)$ complexity for $N$ particles. For instance, when the particles approach

**Algorithm 1** Particle-Particle Method

```
1: For each particle $i$, accumulate the interaction due to each particle $j$ ($i \neq j$) by directly computing the interaction $F(i, j)$.
2: Integrate the equations of motion (using a suitable integrator).
3: Increment the time step.
```

each other, the force between them (and hence the acceleration) would become much larger. Therefore, if we use a constant time step in the integration scheme for calculating the velocity and the position of a particle, the numerical overflow will result in an erroneous simulation output. Thus, we may want to consider a numerical integration scheme that uses variable time steps, instead of a constant time step. Such a scheme should automatically cut down the time step when the particles are near each other and increase the time step when the particles are far away. In most astrophysical simulations, a softening parameter (usually denoted by $\epsilon$) is used to resolve this issue. In spite of having a large computational complexity, PP method has a distinct advantage of being easily parallelizable. Moreover, PP method can be very fast and accurate for a small number of particles (typically $< 1000$) and since it has almost no computational over-head in terms of accessing data structures, it used in conjunction with other methods such as particle-mesh and treecodes.

**Particle-Mesh (PM):** The PM method approximates the force on a mesh by treating it as a field. The system is discretized and finite difference approximations are used for Laplacian operators. The mass (or charge) distribution of the particles is used to assign the densities on the mesh. Force and potential interactions at various particle positions are obtained by interpolations on the assigned densities. The essential steps in a PM method are given in Algorithm 2. One of the basic type of a PM method is the 'Nearest-Grid-Point' (NGP) method where the densities are

**Algorithm 2** Particle-Mesh Method

```
1: Define the densities on the mesh by assigning the values based on the charge or mass distribution.
2: Use a Poisson solver on the mesh.
3: Compute the force field from the mesh-defined potential.
4: Use an interpolation scheme to find the forces on the mesh and assign the forces to the particles.
5: Find the positions and the velocities of the particles by using an integration scheme.
6: Increment the time counter and repeat the process.
```

allocated based on the sum of the magnitude of the charges or masses present in the mesh surrounding the mesh point, divided by the total cell volume. This method is not often used as there is a discontinuity among the forces so obtained.

The main advantage of using a PM method is the speed it offers. The computational complexity of this method is $O(N + M_p \cdot \log(M_p))$, where $M_p$ is the number of mesh points. Another advantage of using this scheme is that integral transforms such as the Fast Fourier Transform or the Wavelet Transform can be applied to speed up the Poisson solver. Also, other numerical techniques such as the finite element method or finite volume method can be used for solving the potential equation. However, mesh-based methods have difficulties in handling non-uniform particle distributions. They offer limited resolution and in such cases finer resolution should be selectively made in the sub-regions to handle non-uniformity.

**Particle-Particle/Particle-Mesh (P3M):** This method resolves the issue of limited resolution in the PM method. In this algorithm, PP method is used over pairs of particles which are closely separated in the mesh spacing. To calculate the far-field forces, the inter-particle forces are broken down into two components: The rapidly varying short range part and the slowly varying long range part. For the short range interactions, PP method is used whereas PM is used to compute the long range interactions.

In P3M, two meshes are used to compute the forces: a mesh for the PM method that has the densities and a mesh that can be used to locate the pairs of neighboring particles that contributes to the short range interaction. In general, the scheme works as shown in Algorithm 3.

**Algorithm 3** Particle-Particle/Particle-Mesh Method

```
1: Assign the densities to the first mesh and calculate the forces through the PM
   method (solve for potentials, interpolate forces, increment momenta) for particles
   that are well-separated.
2: Use the second mesh and find the forces through the PP method for nearby
   particles.
3: Use integrators to get the position and velocity.
4: Increment the time step and repeat the process.
```

This method has been widely used in simulations where the forces are easily split into long range and short range components. Moreover, Fourier Transform can be used to reduce the computation time, however, the main disadvantage of this method is that the direct summations can easily become a dominating factor.

**Ewald Summation:** In electrostatics, the Ewald summation method is extensively used to compute the pairwise long range and short range interactions in periodic systems. Just as in the P3M, this method works by dividing the lattice summation into a short range part and a long range part. The long range-part is evaluated in a fast converging Fourier representation and for short range interaction, PP method is applied. Finally the forces are summed up over all particles in the Fourier space and a self-energy correction is applied. This method is mostly applied to the Gaussian distribution and the accuracy of the method depends on the approximations made both in the real space and Fourier space. The approximations are characterized by the width of the Gaussian charge distribution, the cutoff-radius in the real space part and the maximum wave number in the Fourier space part.

**Treecodes:** Treecodes hierarchically decompose the domain and approximate the potential/force interactions through multipole expansions. They are highly suitable for non-uniform distributions and have no preferred geometry. They can be used for periodic or non-periodic boundary conditions. Treecodes are particularly effective for modelling astrophysical simulations where there could be collisions between the galaxies. This dissertation discusses different types of treecodes for various physical problems. In the next section, we will give an overview of the treecodes and describe the Barnes-Hut method in detail which is often used in the simulation of astrophysical problems.

## 2. Overview of the Treecodes

Treecodes use hierarchical data structures and they provide fast and accurate approximations for the potentials, forces and other interaction laws in $N$-body simulations. The fundamental idea behind these methods is the recognition that the combined potential of a group of particles which are far away can be well-approximated by a low-order multipole series expansion. In general, hierarchical schemes partition the mass (or charge) distribution into a tree structure, where each node of the tree is a representation of the matter within some spatial volume. In three dimensions, the domain is recursively subdivided into eight equal sub-domains and this process is carried out until each sub-domain has at most one element (See Figure 1). Then for

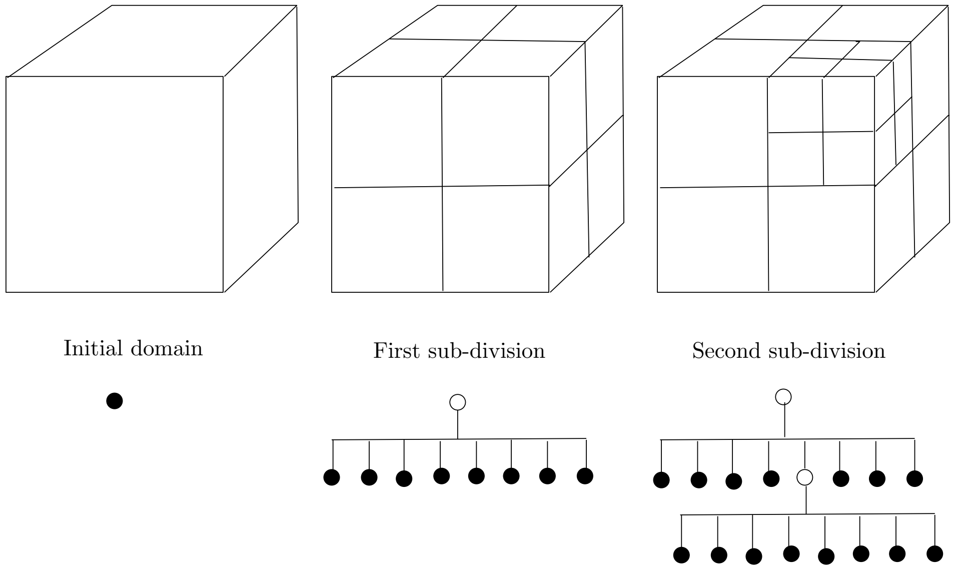

Fig. 1. Recursive division of the domain and oct-tree representation

each node in the tree, multipole moments (typically only up to quadrupole order) are calculated and stored. This representation is used for approximating the interaction law to the desired accuracy. Treecodes are quite popular because the asymptotic scaling is usually $O(N \log N)$ or $O(N)$. As opposed to the PP method that evaluates the potential exactly, in treecodes there is a trade off between accuracy and computational cost, with higher accuracy demanding higher computational cost. Greengard and Rokhlin proposed the Fast Multipole Method (FMM) [7] which is considered to be the fastest contemporary treecode to solve potential evaluation problems involving Coulomb potentials (potentials of the form $r^{-1}$). Barnes-Hut [4] treecodes are used for force calculations. Since our treecodes are similar to the Barnes-Hut treecode, we will illustrate the fundamental idea behind this method.

## 3. The Barnes-Hut Method

Barnes-Hut method is a force approximation technique introduced by Joshua Barnes and Piet Hut in [4]. Like most hierarchical tree based methods, at every time step, an oct-tree is constructed using the domain decomposition as illustrated in Figure 1. Once the tree is constructed the multipole moments are computed and stored for each node in the tree and the force on a particle is computed by traversing down the tree starting from the root node. At each node of the tree, a distance criteria is applied to see if the node is distant enough for a force evaluation and if so, the force is evaluated using the moments. If the criteria is not satisfied, the node is "expanded" and the criteria is applied for its eight children. If there are no children for a node, PP method is used. It can be seen that by this process, the number of interactions computed is significantly smaller than the PP method. The complexity of this method is $O(N \log N)$ as one may traverse at most the height of the tree for each particle. Typically, in a simulation consisting of $10^6$ particles, there could be around 1000 interactions per particle on an average making the algorithm significantly faster than the PP method. The pesudocode for the Barnes-Hut method is given in Algorithm 4. There are different distance criteria that can be applied for determining whether a node is sufficiently far away from a particle for force evaluation. The simplest criterion is based on the ratio of the dimension of the sub-domain and the distance from the particle to the center of mass of the sub-domain. Given the distance of a point from the center of mass of the sub-domain, $r$, and the dimension of the sub-domain, $d$, a point is considered to be far away from a node if,

$$\alpha > \frac{d}{r}, \tag{1.1}$$

**Algorithm 4** Barnes-Hut Method

```
1: Construct the octree until there are at most 1 particle per leaf box.
2: Compute and store multipole moments (or the total mass) for each node in the
   tree. For the leaf nodes compute them directly and for the parent nodes, translate
   the moments (or the total mass) from its child.
3: for each particle do
4:     Apply the distance criteria to the root of the tree and see if an interaction can
       be computed. If not, go to the children of the node and apply the distance
       criteria. If there are no children, compute the interactions directly.
5: end for
6: Add the interactions.
7: Use integrators to get the position and velocity.
8: Increment the time step and repeat the process.
```

where $\alpha$ is a specific user-defined constant. Smaller values of $\alpha$ lead to more accurate force evaluation. Typically, $\alpha = 1$ gives accelerations with errors around 1% [18]. In [19], Salmon & Warren showed that the above criterion for determining the applicability of the force calculation can introduce more errors when the center of mass is near the edge of the sub-domain. They introduced various distance criteria which avoid this problem. One of the criteria is

$$r > \frac{d}{\alpha} + \delta,$$

where $\delta$ is the distance between the center of mass of the sub-domain and its geometric center (see Figure 2). One can see that if the center of mass of the sub-domain is near the geometric center, then $\delta$ becomes small and so this criteria is same as criteria (1.1). This criteria has been found to perform well even when the center of mass is near the

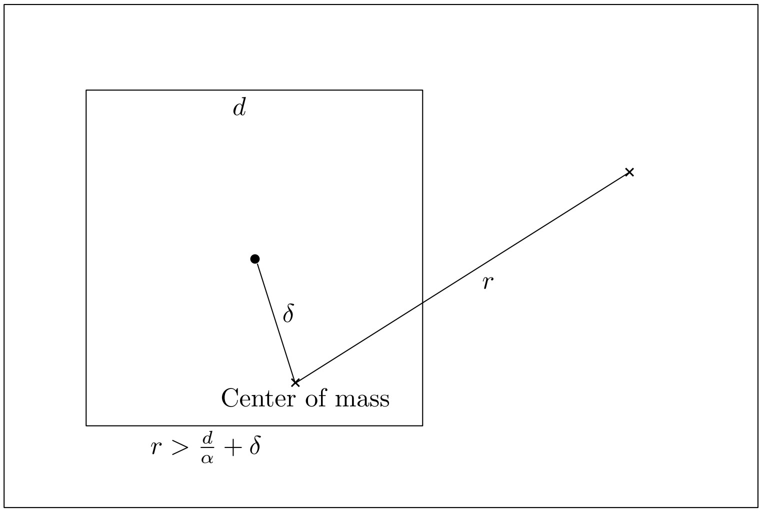

Fig. 2. Geometry of the distance criteria introduced by Salmon et al.

edge of the sub-domain.

### B. Some Limitations of Treecodes

Although treecodes have proved to be of great advantage in reducing the computational complexity in $N$-body simulations, existing methods have a several shortcomings. One of the primary requirement in using a treecode is that the multipole expansions should exist for the kernel function appearing in the interaction law. This is because the applicability of these codes are limited to operators whose multipole expansions are available analytically. Although, there are kernel independent treecodes available [31], their effectiveness in $N$-body simulations haven't been studied yet. In the case of potential evaluations, only few methods have been proposed for problems where the Green's function is of the form $r^{-\lambda}$, $\lambda \geq 1$ since constructing multipole expansions are not straight-forward. It is well acknowledged that computing these potentials is an important task in many fields like molecular dynamics [21], computational chemistry [20] and fluid mechanics [22]. Even through there have been some advances in computing these potentials, current methods employ Cartesian coordinates to evaluate the interactions. The choice of spherical coordinates is essential in these computations because the spherical harmonics basis functions can be employed for efficient computations. For radially symmetric functions like $r^{-\lambda}, \lambda \geq 1$, the use of Cartesian coordinates to separate the variables usually results in cumbersome expressions. Moreover, such a coordinate system does not exploit the inherent spherical symmetry found in these kernels. Duan et al. [23] have proposed treecodes in which multipole approximations are obtained using Taylor series in Cartesian coordinates. Their scheme involves expensive computations ($O(p^6 N \log N)$ for a multipole approximation degree $p$) to find the total potential energy of the system. Chowdhury et al. [24] have used the addition theorem for ultraspherical polynomials in Cartesian coordinates to find far field effects for these potentials. They develop necessary operators needed for a single level multipole method. However, it is unclear how these operators can be effectively used to compute the total potential at a point which includes the near field effects as well.

In the simulations that study the movement of the particles, the shortcoming of treecodes are quite apparent. For instance, typically in a treecode the oct-tree and the multipole moments are re-constructed during every time step irrespective of the distance travelled by the particles. The cost in computing these moments increases with the accuracy and they naturally contribute a dominant share to the overall execution time of the simulation. It may be necessary to compute all the moments at every time step if the particles change their positions rapidly during the time interval. However, in most simulations, the displacements are small, especially in the simulation of collisionless systems with large number of particles (such as the dynamical evolution of a galaxy or in the conformation of molecules). This is because, in such systems, the movement is largely restricted by the average interparticle distance, which is inversely proportional to the number of particles. Thus, if there are large number of particles then they are bound to move only small distances over time steps.

Intuitively, when there is a only a small change in the position of the particles, there may not be a large difference in the computed potentials. Therefore, we can retain the tree structure and re-use the moments computed at some earlier time step for future time advances. In fact, the map that takes a particle's location to its multipole moment is a continuous map and hence the moment should vary a little if there is only a little change in the position. Furthermore, in treecodes, a node's multipole moment will be used only for the evaluation points lying outside its boundary. For a particle with a unit charge and displacement $\delta\rho$, the difference in the potential after the displacement, $\Delta P$, can be shown to be,

$$\Delta P \propto \frac{1}{r-\rho}\left(\frac{\delta\rho}{r-\rho}\right)^{p+1},$$

where $r$ and $\rho$ are the distances from the center of the node to the evaluation point and the particle location, respectively and $p$ is the multipole degree. If $\delta\rho \ll r$ and the ratio $\rho/r$ remains the same for an ancestor node $N_A$, and its descendent $N_D$, we may expect the difference in the potential introduced by $N_A$ to be lesser than $N_D$. This is because the spatial extent of $N_A$ increases at least by a factor of two than $N_D$ and $\left(\frac{\delta\rho}{r-\rho}\right)^{p+1}$ decreases rapidly for a high multipole degree $p$. Consequently, when the particles travel a small distance between time steps, the idea of re-using the moments (especially for the ancestor nodes) and the tree, derived at an earlier time step, looks compelling. The cost of constructing the tree is not as expensive as computing the multipole moments but it can be carried out only when it is necessary. This view has already been suggested by J.Makino [25] where the tree is constructed once in every ten time steps, nevertheless the idea of updating only a few multipole moments remains unaddressed.

## C. Scope of this Dissertation

The scope of this dissertation can be broadly classified into following three domains often considered in $N$-body simulations: (*i*) Computing the potentials of the form $r^{-\lambda}$, (*ii*) Force evaluations in $N$-body simulations, (*iii*) Potential approximations in particle simulations. We develop efficient treecodes that focus on the limitations discussed in Section B. These treecodes are given in Chapters II, III and IV. In Chapter II, we propose a new multipole expansion based treecode for computing the potentials of the form $r^{-\lambda}$ where $\lambda \geq 1$. We compare our method with the existing schemes and show that our's is asymptotically superior. It can be seen that our treecode addresses some of the issues examined in Section B. In Chapter III, we will describe how this expansion series can be used to represent gravitational forces. We use this representation in a treecode to compute the forces to the desired accuracy. We also review some methods that are used for force approximations in $N$-body simulations and show how our tree code addresses the need for accuracy in astrophysical problems. In Chapter IV, we establish an asymptotic bound for the difference in the Coulomb potential when the particles are moving and illustrate the usefulness of this bound by devising an algorithm that runs faster than a standard treecode used in $N$-body simulations. We recall that the current procedures have a draw back that multipole moments are computed at every time step during the simulation irrespective of particle's displacements. We propose a scheme in this chapter that responds to this issue by retaining the multipole moments computed at an earlier time step. In all the chapters we illustrate the effectiveness of our treecodes with several numerical experiments.

# CHAPTER II

# A TREECODE FOR POTENTIALS OF THE FORM $r^{-\lambda}$*

In this chapter we propose a multipole based treecode that uses spherical harmonics to evaluate potentials of the form $r^{-\lambda}$, where $\lambda \geq 1$. The chapter is based on the works presented in [1] and [2]. This treecode has several advantages over existing Cartesian coordinate based expansion schemes discussed in Section B of Chapter I. First, our algorithm has a complexity of $O(p^3 N \log N)$ for a small constant $p$, which is faster than any known methods to solve this problem suitably. Second, the use of spherical harmonics gives an analytic formula to compute multipole coefficients efficiently. This analytic formula is a natural generalization of the multipole expansion theorem for $r^{-1}$ potential. Third, by using spherical coordinates (and spherical harmonics), we make use of the spherical symmetry in $r^{-\lambda}$ kernels to precompute certain vectors that significantly reduce the time to compute multipole coefficients. And finally, it is straightforward to modify an existing FMM implementation to evaluate $r^{-\lambda}$ potentials using this approach.

The chapter is organized as follows: Section A discusses the multipole expansion theorem for $r^{-1}$ functions. Section B gives the multipole expansion theorem for $r^{-\lambda}$ potentials based on the relationship between ultraspherical polynomials and Legendre polynomials. Section C describes a treecode to compute $r^{-\lambda}$ interactions. Section D analyzes the complexity of the algorithm and compares this method with existing algorithms. Section E discusses implementation details and numerical experiments.

---

*Reprinted with permission from "A treecode for potentials of the form $r^{-\lambda}$" by Kasthuri Srinivasan and Vivek Sarin, 2007, International Journal of Computer Mathematics, 84, 1249-1260, Copyright 2007 by Taylor and Francis.

## A. Multipole Expansion Theorem for $r^{-1}$ Potential

This section describes the multipole expansion theorem from the classical FMM [7]. Let $P$ and $Q$ be points with spherical coordinates $(\rho, \alpha, \beta)$ and $(r, \theta, \phi)$, respectively, and $\gamma$ be the angle subtended by them at the origin. If $r'$ is the Euclidean distance between $P$ and $Q$, then for $\rho < r$, the potential $\Phi(P)$ at $P$ due to a charge $q$ at $Q$ is given by

$$\Phi(P) = \frac{q}{r'} = \sum_{n=0}^{\infty} \frac{q \cdot \rho^n}{r^{n+1}} P_n(\cos\gamma), \tag{2.1}$$

where $P_n(\cos\gamma)$ is the Legendre polynomial of degree $n$. We also have the addition theorem for Legendre polynomials

$$P_n(\cos\gamma) = \sum_{m=-n}^{n} Y_n^{-m}(\alpha, \beta) Y_n^m(\theta, \phi), \tag{2.2}$$

where $Y_n^m$ are the spherical harmonics given by

$$Y_n^m(x, y) = \sqrt{\frac{n-|m|}{n+|m|}} P_n^{|m|}(\cos x) e^{imy}, \tag{2.3}$$

in which $P_n^m$ are the associated Legendre functions evaluated at $\cos x$. Using (2.1) and (2.2) we have the following theorem.

**Theorem 1.** *Suppose that a charge of strength $q$ is located at $Q(\rho, \alpha, \beta)$ with $\rho < a$. At any point $P(r, \theta, \phi)$ with $r > a$, the potential $\Phi(P)$ is given by*

$$\Phi(P) = \frac{q}{r'} = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \frac{M_n^m}{r^{n+1}} Y_n^m(\theta, \phi), \tag{2.4}$$

*where $M_n^m = q\rho^n Y_n^{-m}(\alpha, \beta)$.*

If there are several charges $\{q_i : i = 1, \dots k\}$ in a disc of radius $a$ centered at $Q$ with coordinates $\{(\rho_i, \alpha_i, \beta_i) : i = 1, 2, \dots k\}$ we can compute their combined multipole moment at $Q$ as $M_n^m = \sum_{i=0}^{k} q_i \rho_i^n Y_n^{-m}(\alpha_i, \beta_i)$. These coefficients can be used in (2.4) to compute the potential at $P$ due to all the charges. Fig. 3 illustrates the theorem given above for a cluster of charge particles $Q_1, Q_2, Q_3$ and $Q_4$. The coefficients $M_n^m$ are found using only the information on the particle locations and the charge of the particles. No information on the point of evaluation is required to compute the moment. This moment can be independently used for evaluating the potential at a distant point $P$.

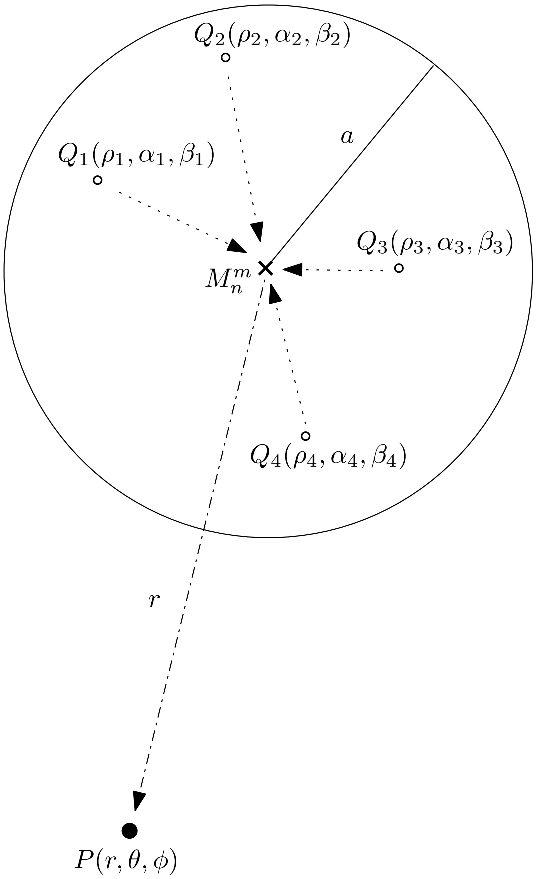
$Q_2(\rho_2, \alpha_2, \beta_2)$
$Q_1(\rho_1, \alpha_1, \beta_1)$
$a$
$Q_3(\rho_3, \alpha_3, \beta_3)$
$M_n^m$
$Q_4(\rho_4, \alpha_4, \beta_4)$
$r$
$P(r, \theta, \phi)$

Fig. 3. Multipole expansion for a cluster of charge particles

## B. Ultraspherical Polynomials

Ultraspherical polynomials (or Gegenbauer polynomials) are a *higher dimensional* generalization of Legendre polynomials [27, 28]. Ultraspherical polynomials are eigenfunctions of the generalized angular momentum operator just as Legendre polynomials are eigenfunctions of the angular momentum operator in three dimensions. Ultraspherical polynomials allow addition theorem using hyperspherical harmonics similar to Legendre polynomials which allow addition theorem using spherical harmonics. One can refer to [27] for a list of similarities between ultraspherical and Legendre polynomials. Ultraspherical polynomials are also a generalization of Legendre polynomials in terms of the underlying generating function. Thus, if $x, y \in \Re$, then

$$\frac{1}{(1-2xy+y^2)^{\lambda/2}} = \sum_{n=0}^{\infty} C_n^{\lambda}(x) y^n, \tag{2.5}$$

where $C_n^{\lambda}(x)$ is the ultraspherical polynomial of degree $n$. This generating function can be used to expand $r^{-\lambda}$ using ultraspherical polynomials. Let $P(r, \theta, \phi)$ and $Q(\rho, \alpha, \beta)$ be points with spherical coordinates. Using (2.5) we have,

$$\frac{1}{(r')^{\lambda}} = \frac{1}{r^{\lambda}(1-2u\mu+\mu^2)^{\lambda/2}} = \sum_{n=0}^{\infty} \frac{\rho^n}{r^{n+\lambda}} C_n^{\lambda}(u), \tag{2.6}$$

where $\mu = \rho/r$ and $u = \cos\gamma$. There are addition theorems for ultraspherical polynomials using hyperspherical harmonics which allow us to separate $C_n^{\lambda}(u)$ in terms of the coordinates. Instead, we use a relation that exists between ultraspherical and Legendre polynomials [29], which allows us to use spherical harmonics in three dimensions. Let $P_n(x)$ and $C_n^{\lambda}(x)$ be Legendre and ultraspherical polynomials of degree $n$, respectively. Then

$$C_n^{\lambda}(x) = \sum_{s=0}^{\lfloor n/2 \rfloor} \frac{(\lambda)_{n-s}(\lambda-1/2)_s}{(3/2)_{n-s}s!}(2n-4s+1)P_{n-2s}(x), \tag{2.7}$$

where $(r)_s = r(r+1)(r+2)\dots(r+s-1)$. We define

$$B^{\lambda}_{n,s} = \frac{(\lambda)_{n-s}(\lambda-1/2)_s}{(3/2)_{n-s}s!}(2n-4s+1). \tag{2.8}$$

We now derive an addition theorem for ultraspherical polynomials. The following addition theorem is found in different form in many places [27, 28] but as far as we know this specific form has not previously appeared.

**Theorem 2. (Addition Theorem for Ultraspherical Polynomials)** *Let $P$ and $Q$ be points with spherical coordinates $(r,\theta,\phi)$ and $(\rho,\alpha,\beta)$, respectively, and let $\gamma$ be the angle subtended by them at the origin. Then*

$$C^{\lambda}_n(\cos\gamma) = \sum_{m=0}^{\lfloor n/2\rfloor} B^{\lambda}_{n,m}\mathbf{Y_{n,m}}(\theta,\phi)\cdot\overline{\mathbf{Y_{n,m}}}(\alpha,\beta),$$

*where $\mathbf{Y^T_{n,m}}(x,y) = [Y^{-(n-2m)}_{n-2m}(x,y), Y^{-(n-2m)+1}_{n-2m}(x,y),\dots,Y^{(n-2m)}_{n-2m}(x,y)]$ is a vector of spherical harmonics of degree $n-2m$.*

*Proof.* The proof follows from (2.7), which relates ultraspherical polynomials and Legendre polynomials, and the addition theorem for Legendre polynomials given by (2.2). □

Once we have an addition theorem for ultraspherical polynomials we can prove the multipole expansion theorem for $r^{-\lambda}$ potentials.

**Theorem 3. (Multipole Expansion)** *Suppose that $k$ charges of strengths $\{q_i, i = 1,\dots k\}$ are located at the points $\{Q_i = (\rho_i,\alpha_i,\beta_i), i = 1,\dots,k\}$, with $|\rho_i| < a$. Then for any point $P = (r,\theta,\phi)$ with $r > a$, the potential $\Phi(P)$ is given by*

$$\Phi(P) = \sum_{n=0}^{\infty}\sum_{m=0}^{\lfloor n/2\rfloor}\frac{1}{r^{n+\lambda}}\mathbf{M^m_n}\cdot\mathbf{Y_{n,m}}(\theta,\phi), \tag{2.9}$$

*where*

$$\mathbf{M_n^m} = \sum_{i=1}^{k} q_i \rho_i^n B_{n,m}^{\lambda} \overline{\mathbf{Y_{n,m}}}(\alpha_i, \beta_i). \tag{2.10}$$

*Furthermore, for any* $p \geq 1$,

$$|\Phi(P) - \sum_{n=0}^{p} \sum_{m=0}^{\lfloor n/2 \rfloor} \frac{\mathbf{M_n^m}}{r^{n+\lambda}} \cdot \mathbf{Y_{n,m}}(\theta, \phi)| \leq \frac{AB_\lambda}{r^{\lambda-1}(r-a)} \left(\frac{a}{r}\right)^{p+1}, \tag{2.11}$$

*where* $A = \sum_{i=1}^{k} |q_i|$ *and* $B_\lambda = \max\limits_{0 \leq n \leq p} \left\{ \sum_{m=0}^{\lfloor n/2 \rfloor} |B_{n,m}^{\lambda}| \right\}$.

*Proof.* Suppose $\Phi_i(P)$ is the potential at $P$ due to $q_i$ at $Q_i$. From (2.6) and Theorem 2, we have

$$\begin{aligned} \Phi_i(P) &= \sum_{n=0}^{\infty} \frac{q_i \rho_i^n}{r^{n+\lambda}} C_n^{\lambda}(u) \\ &= \sum_{n=0}^{\infty} \sum_{m=0}^{\lfloor n/2 \rfloor} \frac{\left[q_i \rho_i^n B_{n,m}^{\lambda} \overline{\mathbf{Y_{n,m}}}(\alpha_i, \beta_i)\right]}{r^{n+\lambda}} \cdot \mathbf{Y_{n,m}}(\theta, \phi). \end{aligned}$$

The moments in (2.10) are obtained by superposition.

We now prove the error bound. Note that for every $u \in \Re$ with $|u| \leq 1$, we have $|P_n(u)| \leq 1$. From (2.7) using the triangle inequality it can be seen that,

$$|C_n^{\lambda}(u)| \leq \sum_{m=0}^{\lfloor n/2 \rfloor} |B_{n,m}^{\lambda}| \leq \max_{0 \leq n \leq p} \left\{ \sum_{m=0}^{\lfloor n/2 \rfloor} |B_{n,m}^{\lambda}| \right\} = B_\lambda.$$

Now, for each $q_i$ located at $Q_i$ having $|\rho_i| < a$, we have,

$$\begin{aligned} |\Phi_i(P) - \sum_{n=0}^{p} \frac{q_i \rho_n^n}{r^{n+\lambda}} C_n^{\lambda}(u)| &= |\sum_{n=p+1}^{\infty} \frac{q_i \rho_i^n}{r^{n+\lambda}} C_n^{\lambda}(u)| \\ &\leq \frac{B_\lambda q_i}{r^\lambda} \sum_{n=p+1}^{\infty} \left(\frac{\rho_i}{r}\right)^n \\ &\leq \frac{B_\lambda}{r^{\lambda-1}} \frac{q_i}{r-a} \left(\frac{a}{r}\right)^{p+1}. \end{aligned} \tag{2.12}$$

The error bound (2.11) is obtained by superposition of error bounds for all the $k$

charges. □

## C. A Treecode for $r^{-\lambda}$ Potentials

The treecode is similar to the one that was described in Chapter I (refer Sections 2 and 3), except that it differs in the way we compute the multipole moments. First, an oct-tree representation of the domain is obtained by recursively sub-dividing it into octants. This procedure is carried out until each domain has at most one particle. After this process, for each node we compute the multipole series representation. We do this using the relative position of the particle within the node with respect to its geometric center. This is different from the Barnes-Hut treecode (see Algorithm 4) where the multipole moments are computed using a child-parent translation operator for the parent nodes and calculated directly for the leaf nodes. The use of translation operators, although saves time, causes additional errors. Since we are primarily interested in the error aspect of these treecodes, we avoid the translations and focus on "translation-free" treecodes here and in subsequent chapters.

Once the tree is constructed and the multipole moments are calculated, a distance criteria called the *multipole acceptance criterion* ($MAC$) is applied to compute the potential. Given the distance of a point from the center the sub-domain $d$, and the dimension of the sub-domain $r$, a point is considered to be far away from a node if the ratio $d/r$ is greater than $\alpha$, a user-defined constant. We note that this multipole acceptance criteria is same as the initial distance criteria discussed in Chapter I, Section 3. Thus, the Barnes-Hut method is applied for the potential computation phase.

## D. Analysis of the Algorithm

### 1. Complexity

We now analyze the complexity of our algorithm. It can be seen from the multipole expansion theorem that the complexity for computing the multipole coefficients at each level of the tree is $O(p^3N)$, where $N$ is the number of particles in the system and $p$ is the multipole degree. For uniform particle distributions there are $\log N$ levels in the tree, and therefore, the total cost of computing the multipole coefficients is $O(p^3N \log N)$. Similarly, the complexity for the potential evaluation phase is $O(p^3N \log N)$. Thus, the overall complexity for the algorithm is $O(p^3N \log N)$. Since we are using an oct-tree, the base of the logarithm is 8.

### 2. Comparisons with Other Methods

We now compare our treecode with two relevant methods for the potential evaluation problem. Duan et al. [23] have proposed an algorithm which calculates the total potential energy of a system for $r^{-\lambda}$ potentials. Their scheme can also be used to find the potential at individual particles. Another pertinent method is due to Chowdhury et al. [24]. They have developed single level translation operators for these potentials to calculate far field interactions in particle systems.

We first consider Duan et al. approach. Their method can be summarized as follows: Given $N$ particles, a spatial tree representation of the domain containing the particles is derived. In this phase, the domain is recursively divided into equal sub-domains until there are at most specified number of particles in any box. The total potential energy of the system can now be calculated by adding the potential energy of each box. The potential energy of any box is evaluated either directly or computed using multipole approximations. If a node is a leaf, then the energy of the box is calculated using particle-particle interactions. If not, the node is expanded to see whether the multipole approximations can be used to find the energy for each of its sub-domain. If a sub-domain $A$ is sufficiently far away from another sub-domain $B$, then the multipole approximation can be used to find the total potential of $A$ due to all particles in $B$. Otherwise, the node representing the sub-domain $B$ is further expanded to check whether the children are far away from $A$ so that the approximations can be applied. The procedure recursively proceeds starting from the root.

It can be seen that this cluster-cluster algorithm can be used to find the potential of an individual particle by allowing a cluster to be composed of a single particle. The resulting particle-cluster algorithm will scale to $O(cN \log N)$, where $c$ depends on the multipole approximation degree being used. The authors use multivariate Taylor series expansion in Cartesian coordinates to find multipole approximations. Using Cartesian coordinates, it takes $O(p^6)$ operations to find the interactions between two clusters where $p$ is the pre-specified multipole approximation degree. Therefore, their algorithm has a complexity of $O(p^6 N \log N)$. This high cost for large values $p$ is a result of using Taylor series approximation in Cartesian coordinates. This can be compared with the complexity of our approach where the spherical symmetry of $r^{-\lambda}$ potentials when expressed in spherical coordinates lead to efficient computations. Further, the decreasing nature of the indices in the Legendre polynomial's relation with the ultraspherical polynomial in (2.7), allows us to use the spherical harmonics basis functions in three dimensions instead of computing vector of spherical harmonics given in Theorem (2). Thus, computing $O(p^2)$ spherical harmonics for each particle is enough to compute $O(p^3)$ multipole coefficients which is a significant reduction in time for large particle systems. Another advantage in using our algorithm is that an existing FMM implementation can be easily modified to compute $r^{-\lambda}$ potentials.

Also, we note that, using our approach we can find the total potential of the system by summing individual particle's potential.

We now consider Chowdhury et al. [24]. The authors derive expressions for translation operators for these potentials. These operators can be used to evaluate the far field effects of the potential by combining with a single level translation scheme. A single level translation based algorithm (usually called $SLFMM$ to denote Single Level Fast Multipole Method) using such operators was proposed in [30]. $SLFMM$ works in three phases. In the first phase, the domain consisting of $N$ particles is grouped into clusters and multipole coefficients are computed and stored for each cluster. In the second phase, the multipole coefficients from each cluster are translated to every other cluster by means of a translation operator and they are converted into local coefficients. In the third phase, these local coefficients are used to find the far field effects which are evaluated at each particle. The near field potentials are computed directly. It can be seen that for an approximation degree $p$, the complexity for $SLFMM$ is,

$$\text{Complexity}_{\text{slfmm}} = O\left(\frac{N^2}{K} + K^2 * \text{TransCost(p)} + N * \text{MultipoleCost(p)}\right),$$

where $K$ denote the number of clusters, TransCost(p) denote the number of operations per translation and MultipoleCost(p) is the cost for finding the multipole coefficients in any cluster. The $\frac{N^2}{K}$ term in the complexity is the number of direct interactions computed. Minimizing the complexity with respect to the number of clusters, we find the optimum number of clusters, $K_{opt}$, is given by,

$$K_{opt} = \left[\frac{N^2}{2 * \text{TransCost(p)}}\right]^{1/3}.$$

Therefore, the complexity of $SLFMM$ with optimum choice of $K$ is,

$$\text{Complexity}_{\text{opt}} = O(N^{4/3} * \text{TransCost}^{1/3}(\text{p}) + N * \text{MultipoleCost}(\text{p})). \tag{2.13}$$

In case of [24], each translation operation costs $O(p^6)$ and the cost of finding the multipole coefficients is $O(p^3 N)$. For $N^{1/3} > p$, if we consider the principal term in (2.13), the complexity of $SLFMM$ when used with operators given in [24], is $O(p^2 N^{4/3})$. We compare this with the complexity of our algorithm which is $O(p^3 N \log_8 N)$. Since, $\log_8 N < N^{1/3}/p$ for sufficiently large values of $N$, we see that our algorithm will outperform $SLFMM$ with translation operators defined by Chowdhury et al.

## E. Numerical Experiments

The algorithm was coded in C++ programming language and the computations were performed in double precision on a 2.4 GHz, 512 MB Intel P4 PC running Red Hat 9.0. The code first constructs the oct-tree and the multipole coefficients are calculated along with the tree construction. Standard Template Library (STL) data structures were used for efficient memory and time management for the dynamically adaptive tree construction phase. For a given $\lambda$, the constants defined in (2.8) were precomputed and used in the potential evaluation phase to reduce the overall computation time. Furthermore, for every point we compute only $O(p^2)$ spherical harmonics to calculate $O(p^3)$ multipole coefficients. We conduct three experiments to study the behavior of the treecode. In all the experiments, a direct summation code is the benchmark for comparing errors and execution time. In the first experiment we use the Coulomb potential ($\lambda = 1$) to study the dependence of the execution time on the system size. In the second experiment we analyze the accuracy and the execution time by varying the multipole degree and the $MAC$ constant. For the third experiment, we use the London dispersion potential ($\lambda = 6$) to compare the accuracy with the Coulomb potential. The parameters in the experiments are: multipole degree $p$, $MAC$ constant $\alpha$, maximum number of particles in a leaf box $s$, and the total number of particles in the system $N$. The magnitude of the charges are fixed at $+1$. Random particle distributions on a cube of unit length are used for all the experiments. To measure accuracy, we use the *root mean square relative error (RMS)* defined by

$$RMS = \sqrt{\frac{1}{N}\sum_i \left[\frac{|\Phi_{tree}(i) - \Phi_{direct}(i)|}{|\Phi_{direct}(i)|}\right]^2}, \tag{2.14}$$

where $\Phi_{tree}(i)$ is the potential for the $i^{th}$ particle obtained using the treecode and $\Phi_{direct}(i)$ is the potential of the $i^{th}$ particle computed using the direct summation algorithm.

### **The Coulomb Potential** ($\lambda = 1$)

Figure 4 shows the dependence of the execution time on the system size for the

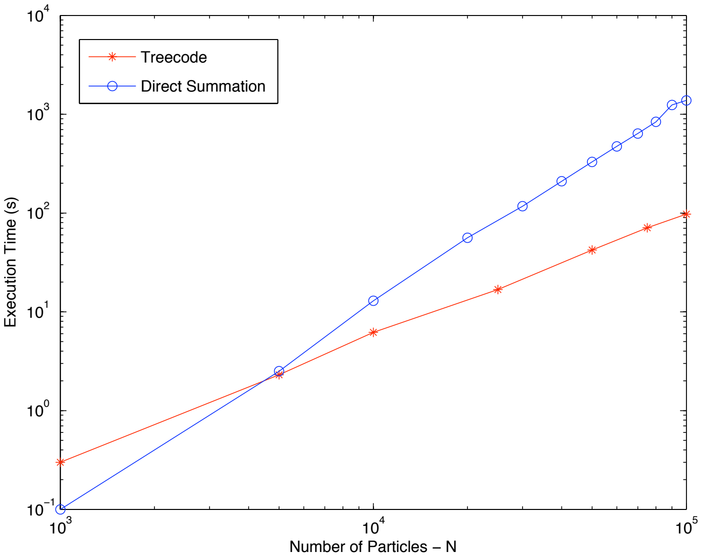

Fig. 4. System size Vs execution time

specified potential function. Here the system size varies from $N = 10^3$ to $N = 10^5$. The multipole degree $p$ is fixed at 2, the maximum number of particles in the leaf box is 10 and the $MAC$ constant is fixed at 1.4. We observe that the execution time of the treecode is $cp^3 N \log N$ where $c = 0.11$, as determined from the slope of the curve in Figure 4. Tree-based potential evaluation schemes are quite inefficient for small number of particles because the overhead involved in manipulating and traversing the tree data structure is fairly large. This is precisely the reason why the treecode outperforms the execution time of the direct summation algorithm for $N > 5000$.

Table I. Number of particles in the leaf box Vs execution time (in secs)

| | Number of Particles ($N$) | | | | | | |
|---|---|---|---|---|---|---|---|
| $s$ | $1 \times 10^3$ | $5 \times 10^3$ | $1 \times 10^4$ | $2.5 \times 10^4$ | $5 \times 10^4$ | $7.5 \times 10^4$ | $1 \times 10^5$ |
| 10 | 0.3 | 2.3 | 6.2 | 16.8 | 42.2 | 70.8 | 97.5 |
| 25 | 0.1 | 1.9 | 4.3 | 16.5 | 34.2 | 53.8 | 82.1 |
| 50 | 0.1 | 1.9 | 4.0 | 13.6 | 34.3 | 52.8 | 73.3 |

Table I shows the execution time for varying system size where the number of particles in the leaf box $s$ equals 10, 25 and 50. We observe that the execution time decreases as we increase $s$. Note that the tree height decreases when $s$ is increased. Thus, the time to use the $MAC$ to evaluate the potential decreases. Although there will be a slight increase in the time to evaluate direct interactions, this will be negligible because the direct summation time is less for small number of particles (as Figure 4 shows). Hence, the over all execution time decreases. However, it is not advisable to use a large value for the number of particles in the leaf box as this may increase the execution time for direct evaluations substantially.

In Tables II and III, we vary the MAC constant $\alpha$ and plot the execution time and error as a function of the multipole degree $p$. The number of particles in the system is fixed at $10,000$. The $MAC$ constant ranges from $\alpha = 1.11$ to $\alpha = 2.5$ and the maximum number of particles in the leaf box is fixed at 10.

Table II. Execution time (in secs) for the Coulomb potential

| | $MAC$ Constant ($\alpha$) | | | | | |
|---|---|---|---|---|---|---|
| $p$ | 0.9 | 0.8 | 0.7 | 0.6 | 0.5 | 0.4 |
| 2 | 3.5 | 4.6 | 6.2 | 8.9 | 13.1 | 21.4 |
| 3 | 6.3 | 8.3 | 10.8 | 15.5 | 22.9 | 35.9 |
| 4 | 9.3 | 12.1 | 16.2 | 23.4 | 34.1 | 53.5 |

Table III. $RMS$ error (in percentage) for the Coulomb potential

| | $MAC$ Constant ($\alpha$) | | | | | |
|---|---|---|---|---|---|---|
| $p$ | 0.9 | 0.8 | 0.7 | 0.6 | 0.5 | 0.4 |
| 2 | $5.8 \times 10^{-2}$ | $3.6 \times 10^{-2}$ | $2.0 \times 10^{-2}$ | $9.2 \times 10^{-3}$ | $4.8 \times 10^{-3}$ | $2.2 \times 10^{-3}$ |
| 3 | $5.7 \times 10^{-2}$ | $3.5 \times 10^{-2}$ | $1.8 \times 10^{-2}$ | $5.6 \times 10^{-3}$ | $2.4 \times 10^{-3}$ | $8.0 \times 10^{-4}$ |
| 4 | $8.0 \times 10^{-3}$ | $4.3 \times 10^{-3}$ | $2.3 \times 10^{-3}$ | $1.0 \times 10^{-3}$ | $3.0 \times 10^{-4}$ | $1.0 \times 10^{-4}$ |

From Tables II and III, it can be observed that for a reasonable accuracy ($10^{-2}$ to $10^{-3}$ % rms error), increasing the $MAC$ constant is a better choice than increasing the multipole degree. Even with $MAC$ constant as high as 1.66 and multipole degree 2, the execution time is one third as that of the case when the multipole degree is 4. Moreover, the increase in the execution time seems to vary linearly with the $MAC$ constant whereas it nearly doubles with the multipole degree. By increasing the multipole degree we increase the computation time of the multipole coefficients whereas increasing the $MAC$ constant increases the time for the direct interactions.

Therefore, in the overall computation time, if the time for the direct interactions in the potential evaluation phase of the algorithm is very small compared to the time taken to compute the multipole coefficients, one should increase the $MAC$ constant to take more direct interactions. This will not cause any dramatic increase in the execution time (as time for direct interactions is small) and error will be reduced. On the other hand, if the execution time for the direct interaction and multipole coefficients are comparable, then increasing the multipole degree will be useful, because increasing the $MAC$ constant would increase the time for direct interactions even more, which may be expensive for large systems.

**The London Dispersion Potential** ($\lambda = 6$)

We now analyze the accuracy for the London dispersion potential. As in Table III, the number of particles in the system is fixed at $10,000$, the $MAC$ constant varies between 1.11 and 2.50 and the maximum number of particles in the leaf box is 10.

Table IV. $RMS$ error (in percentage) for the London dispersion potential

| | $MAC$ Constant ($\alpha$) | | | | | |
|---|---|---|---|---|---|---|
| $p$ | 0.9 | 0.8 | 0.7 | 0.6 | 0.5 | 0.4 |
| 2 | $6.4 \times 10^{0}$ | $2.8 \times 10^{0}$ | $1.0 \times 10^{0}$ | $3.3 \times 10^{-1}$ | $7.1 \times 10^{-2}$ | $1.2 \times 10^{-2}$ |
| 3 | $5.9 \times 10^{0}$ | $2.4 \times 10^{0}$ | $8.8 \times 10^{-1}$ | $2.8 \times 10^{-1}$ | $5.7 \times 10^{-2}$ | $1.0 \times 10^{-2}$ |
| 4 | $4.0 \times 10^{0}$ | $1.5 \times 10^{0}$ | $4.4 \times 10^{-1}$ | $1.1 \times 10^{-1}$ | $1.7 \times 10^{-2}$ | $1.6 \times 10^{-3}$ |

We studied the execution time for this potential and noticed the behavior was similar to the Coulomb potential (Table II). This is expected because the execution time doesn't directly depend on $\lambda$. We also observe an increase in the rms error compared to the Coulomb potential (see Table IV). To understand this, observe that the relative error in potential at $P(r, \theta, \phi)$ due to a particle at $Q_i(\rho_i, \alpha_i, \beta_i)$ is,

$$\frac{|\Phi_{tree}(r) - \Phi_{direct}(r)|}{|\Phi_{direct}(r)|} \leq \left[\frac{q_i B_\lambda}{r^{\lambda-1}(r-a)}\left(\frac{a}{r}\right)^{p+1}\right]\left[\frac{q_i}{r^\lambda}\right]^{-1} \tag{2.15}$$

$$= \frac{B_\lambda}{1-\frac{a}{r}}\left(\frac{a}{r}\right)^{p+1} \tag{2.16}$$

where $\rho_i < a < r$ and $a/r < 1$. The above bound follows directly from (2.12). Note that $a$ is the size of the largest box containing the particle that does not contain $P$. This implies that $\frac{a}{r}$ is a constant that depends on $MAC$. Since the rms error defined in (2.14) is the root mean square of the relative errors for all the particles, we can expect an increase in the rms error as $B_\lambda$ increases. In Figure 5, we plot the values of $B_\lambda$ as a function of $\lambda$.

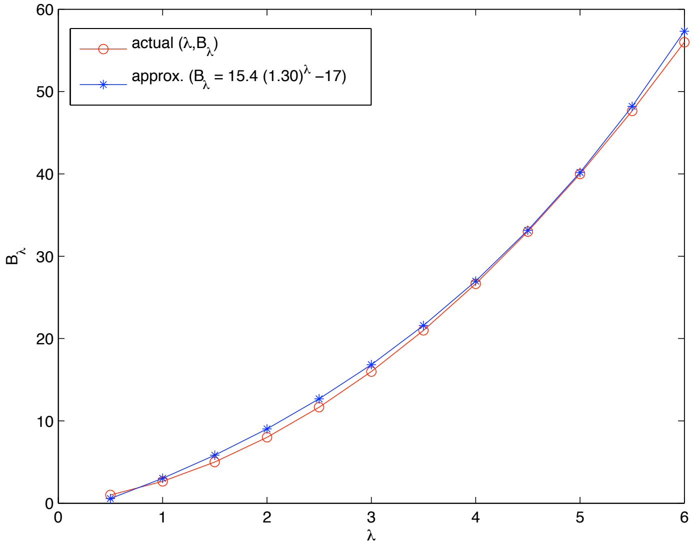

Fig. 5. $\lambda$ Vs $B_\lambda$

We observe that $\lambda$ and $B_\lambda$ are related through the equation $B_\lambda = ab^\lambda + c$, where $a = 15.4$, $b = 1.30$ and $c = -17$ (see Figure 5). This rapid increase in $B_\lambda$ when $\lambda$ increases contributes to the growth in the rms error for the London dispersion potential. We see that for these potentials we have at least 10% increase in the error compared to Table III. Therefore, we may require an increase in the multipole degree or $MAC$ constant to reduce the percentage rms error to $10^{-2}$. This will naturally cause an increase in the execution time.

# CHAPTER III

# AN ACCURATE FORCE CALCULATION ALGORITHM

In this chapter, we extend the treecode introduced in Chapter II to evaluate the forces. This chapter is based on the work presented in [3].

## A. Introduction to Force Approximations

Computing the force interaction between the particles is an integral part of almost all $N$-body codes. For two particles of strengths $q_1$ and $q_2$, the force acting on $q_1$ due to $q_2$, is given by the inverse square law,

$$\vec{F} = \frac{q_1 q_2}{|\boldsymbol{r_{12}}|^2} \cdot \frac{\boldsymbol{r_{12}}}{|\boldsymbol{r_{12}}|}, \tag{3.1}$$

where $|\vec{r_{12}}|$ is the distance between the particles. In this chapter, the constant of proportionality (like the gravitational constant) is assumed to be 1. Even though the above law is applicable to both gravitational and electrostatic interactions, the accuracy determines the algorithms to be used. The Barnes-Hut algorithm described in Chapter I is an easy-to-code algorithm used in most astrophysical simulations where lower accuracy is often preferred (around 1%) over reduced execution time. In astrophysical simulations, the force is approximated by a multipole series expansion with respect to the center of mass. The first moments are large enough (since the masses are all positive) to approximate the forces. Therefore, the monopole moment and the first two moments in the multipole series provide enough accuracy. On the other hand, in MD simulations, higher accuracy is needed because of the distance and time scales involved are very different from astrophysical problems. Barnes-Hut scheme can be adapted to MD simulations by adding more terms to the multipole series. For electrostatic force simulations, however, the distribution of positive and negative charges impedes the numerical robustness of the Barnes-Hut algorithm [8]. Methods like Ewald Summation, particle-mesh based approaches like P3M and FMM are preferred over Barnes-Hut algorithm for electrostatic problems.

In contrast, in the investigation of collisional astrophysical systems, namely star clusters and galactic nuclei, there is a need for high accuracy. Such applications require $|\delta E/E| \ll 0.04N^{-1}t/t_{cr}$ where $t$ and $t_{cr}$ are time parameters, $|\delta E/E|$ is the relative energy accuracy and $N$ is the number of stars in the system [6]. For a system of $10^3$ stars, this means an accuracy of $10^{-5}$. In order to achieve such high accuracy, McMillan and Aarseth [5] use octupole terms in the multipole expansion which are quite cumbersome to construct. Moreover, increased accuracy is needed for larger systems in order to limit the cumulative errors on core-collapse time scales to acceptable levels. To our knowledge there hasn't been any work done which uses higher order moments than octupoles.

In the case of electrostatic force interactions, particle-mesh algorithms like P3M and Ewald Summation techniques compete with FMM [11], [12]. Despite the superior asymptotic scaling of FMM when compared to former methods [$O(N)$ versus $O(N \log N)$], FMM has some disadvantages. Apart from the obvious difficulty in coding, FMM suffers from lack of energy conservation unless very high accuracy is employed [13]. In addition, special considerations regarding the degree of force interpolation must be taken into account to conserve momentum. Pollock and Glosi [12] discuss other advantages of P3M over FMM. Even in force interpolation methods used in particle-mesh algorithms and Ewald Summation methods, a general statement of accuracy seems to be difficult. The error analysis of force calculations in these methods are quite involved and they provide a fair comparison only if done at the same level of accuracy. The discretization methods introduce new sources of errors in addition to the ones originating from real and reciprocal space cutoffs. Furthermore, investigation of errors during force evaluation depends on several parameters such as mesh-size, interpolation orders and differentiation schemes. Deserno and Holm [14] illustrate remarkable differences in accuracy for these methods.

In this chapter, we present a novel algorithm for accurate force calculations in $N$-body problems. This algorithm combines the ease of the Barnes-Hut method with a very high accuracy. Instead of using higher order moments, the algorithm uses additional terms of a multipole series to increase the accuracy. Our method can effectively be used in the calculation of star cluster interactions in which accuracy has been limited by octupole moments. Unlike particle-mesh and Ewald Summation based methods, our algorithm does not involve differentiation (either differentiation in Fourier space, analytic differentiation or discrete differentiation) to calculate the forces. Forces are directly computed through the series. This is especially useful where force interpolation is affected by conservation laws. Further, it may be of interest to note that the complexity of this algorithm is $O(p^3 N \log N)$ which is comparable to that of particle-mesh algorithms that use Fast Fourier Transforms.

The chapter is organized as follows: Section B discusses the multipole expansions for forces calculations. Section C describes the algorithm, analyzes its complexity and discusses implementation issues. Section D presents numerical experiments.

## B. An Expansion Theorem for Forces

We now present an overview of the central idea discussed in Chapter II, i.e., finding the multipole expansion for $r^{-\lambda}$ potentials. The reader is advised to refer to that chapter for further details. The key idea in computing the potentials of the form $r^{-\lambda}$ is the use of ultraspherical polynomials in a manner analogous to the use of Legendre polynomials for the expansion of the Coulomb potential $r^{-1}$. Ultraspherical polynomials (or Gegenbauer polynomials) are generalizations of the Legendre polynomials in terms of the underlying generating function. That is, if $x, y \in \Re$, then

$$\frac{1}{(1-2xy+y^2)^{\lambda/2}} = \sum_{n=0}^{\infty} C_n^{\lambda}(x) y^n, \tag{3.2}$$

where $C_n^{\lambda}(x)$ is a ultraspherical polynomial of degree $n$. They are also *higher dimensional* generalizations of Legendre polynomials [16, 17] in the sense that ultraspherical polynomials are eigenfunctions of the generalized angular momentum operator just as Legendre polynomials are eigenfunctions of the angular momentum operator in three dimensions. One can refer to [16] for a list of similarities between them. The above generating function can be used to expand $r^{-\lambda}$. Let $P(r, \theta, \phi)$ and $Q(\rho, \alpha, \beta)$ be points in spherical coordinate system and $r'$ be the Euclidean distance between them. Using (3.2) we have,

$$\frac{1}{(r')^{\lambda}} = \frac{1}{r^{\lambda}(1-2u\mu+\mu^2)^{\lambda/2}} = \sum_{n=0}^{\infty} \frac{\rho^n}{r^{n+\lambda}} C_n^{\lambda}(u), \tag{3.3}$$

where $\mu = \rho/r$ and $u = \cos\gamma$. There are addition theorems for ultraspherical polynomials using hyperspherical harmonics which allow us to separate $C_n^{\lambda}(u)$ in terms of the coordinates. Due to the complexity involved, we prefer to use a different relation that exists between ultraspherical and Legendre polynomials [15] which allows us to use spherical harmonics in three dimensions. Let $P_n(x)$ and $C_n^{\lambda}(x)$ be Legendre and ultraspherical polynomials respectively, of degree, $n$. Then

$$C_n^{\lambda}(x) = \sum_{s=0}^{\lfloor n/2 \rfloor} \frac{(\lambda)_{n-s}(\lambda-1/2)_s}{(3/2)_{n-s} s!} (2n-4s+1) P_{n-2s}(x), \tag{3.4}$$

where $(p)_s = p(p+1)(p+2)\ldots(p+s-1)$. We define

$$B_{n,s}^{\lambda} = \frac{(\lambda)_{n-s}(\lambda-1/2)_s}{(3/2)_{n-s} s!} (2n-4s+1). \tag{3.5}$$

Using (3.3) and (3.4), we derive the following multipole expansion theorem for $r^{-\lambda}$ potentials.

**Theorem 4. (Multipole Expansion for Potentials)** [**1**] *Suppose that* $k$ *charges of strengths* $\{q_i, i = 1, \dots k\}$ *are located at the points* $\{Q_i = (\rho_i, \alpha_i, \beta_i), i = 1, \dots, k\}$, *with* $|\rho_i| < a$. *Then for any point* $P = (r, \theta, \phi)$ *with* $r > a$, *the potential* $\Phi(P)$ *is given by*

$$\Phi(P) = \sum_{n=0}^{\infty} \sum_{m=0}^{\lfloor n/2 \rfloor} \frac{1}{r^{n+\lambda}} \mathbf{M_n^m} \cdot \mathbf{Y_{n,m}}(\theta, \phi), \tag{3.6}$$

*where*

$$\mathbf{M_n^m} = \sum_{i=1}^{k} q_i \rho_i^n B_{n,m}^{\lambda} \overline{\mathbf{Y_{n,m}}}(\alpha_i, \beta_i)$$

*and*

$$\mathbf{Y_{n,m}^T}(x, y) = [Y_{n-2m}^{-(n-2m)}(x, y), Y_{n-2m}^{-(n-2m)+1}(x, y), \dots, Y_{n-2m}^{(n-2m)}(x, y)]$$

*is a vector of spherical harmonics of degree* $n - 2m$. *Furthermore, for any* $p \geq 1$,

$$|\Phi(P) - \sum_{n=0}^{p} \sum_{m=0}^{\lfloor n/2 \rfloor} \frac{\mathbf{M_n^m}}{r^{n+\lambda}} \cdot \mathbf{Y_{n,m}}(\theta, \phi)| \leq \frac{AB_\lambda}{r^{\lambda-1}(r-a)} \left(\frac{a}{r}\right)^{p+1}, \tag{3.7}$$

*where* $A = \sum_{i=1}^{k} |q_i|$ *and* $B_\lambda = \max_{0 \leq n \leq p} \left\{ \sum_{m=0}^{\lfloor n/2 \rfloor} |B_{n,m}^{\lambda}| \right\}$.

The multipole expansion for force computations can now be deduced as a corollary to the above theorem.

**Corollary 5. (Multipole Expansion for Forces)** *Suppose that* $k$ *particles of strengths* $\{q_i,\, i = 1, \dots k\}$ *are located at the points whose position vectors are* $\{\boldsymbol{\rho}_i, i = 1, \dots, k\}$ *and let* $\{Q_i = (|\boldsymbol{\rho}_i|, \alpha_i, \beta_i), i = 1, \dots, k\}$ *denote their spherical coordinates with* $|\boldsymbol{\rho}_i| < a$. *Then, for any vector* $\boldsymbol{r}$ *with coordinates* $P = (|\boldsymbol{r}|, \theta, \phi)$ *and* $|\boldsymbol{r}| > a$, *the* *force,* $\boldsymbol{F(P)}$*, is given by*

$$\boldsymbol{F(P)} = \sum_{n=0}^{\infty} \sum_{m=0}^{\lfloor n/2 \rfloor} \frac{1}{|\boldsymbol{r}|^{n+3}} \left[\boldsymbol{r} \otimes \mathbf{M_n^m} - \mathbf{V_n^m}\right] \cdot \mathbf{Y_{n,m}}(\theta, \phi) \tag{3.8}$$

*where*

$$\mathbf{M_n^m} = \sum_{i=1}^{k} q_i |\boldsymbol{\rho_i}|^n B^3_{n,m} \overline{\mathbf{Y_{n,m}}}(\alpha_i, \beta_i) \quad \& \quad \mathbf{V_n^m} = \sum_{i=1}^{k} \boldsymbol{\rho}_i \otimes \left(q_i |\boldsymbol{\rho_i}|^n B^3_{n,m} \overline{\mathbf{Y_{n,m}}}(\alpha_i, \beta_i)\right). \tag{3.9}$$

*Here* $\otimes$ *denotes the outer-product. Furthermore, for any* $p \geq 1$*, the approximation*

$$\boldsymbol{\hat{F}_p(P)} = \sum_{n=0}^{p} \sum_{m=0}^{\lfloor n/2 \rfloor} \frac{1}{|\boldsymbol{r}|^{n+3}} \left[\boldsymbol{r} \otimes \mathbf{M_n^m} - \mathbf{V_n^m}\right] \cdot \mathbf{Y_{n,m}}(\theta, \phi)$$

*satisfies*

$$\|\boldsymbol{F(P)} - \boldsymbol{\hat{F}_p(P)}\|_2 \leq \frac{AB_\lambda}{|\boldsymbol{r}|^2(|\boldsymbol{r}| - a)} \left(\frac{a}{|\boldsymbol{r}|}\right)^{p+1} (|\boldsymbol{r}| + a),$$

*where* $A$ *and* $B_\lambda$ *are as defined in (3.7).*

*Proof.* For a particle of strength $q_1$ having spherical coordinate $(|\boldsymbol{\rho}_1|, \alpha_1, \beta_1)$ with $|\boldsymbol{\rho}_1| < a$, the force at any other particle of unit strength having coordinates $(|\boldsymbol{r}|, \theta, \phi)$ with $|\boldsymbol{r}| > a$, is given by (3.1). We note that $\boldsymbol{r_{12}} = \boldsymbol{r} - \boldsymbol{\rho}_1$ (See Fig. 6) and therefore (3.1) can be written as

$$\boldsymbol{F(P)} = \frac{q_1}{|\boldsymbol{r_{12}}|^3}\boldsymbol{r} - \frac{q_1}{|\boldsymbol{r_{12}}|^3}\boldsymbol{\rho}_1.$$

Using $\lambda = 3$ in (3.6) for $1/|\boldsymbol{r_{12}}|^3$, the proof follows through the superposition of particles at $\{\boldsymbol{\rho}_i, i = 1, \ldots, k\}$. The error bound can be obtained from (3.7) along with the triangle inequality $|\boldsymbol{r_{12}}| \leq |\boldsymbol{r}| + |\boldsymbol{\rho}_1| \leq |\boldsymbol{r}| + a$. □

Fig. 7 illustrates the theorem given above for a cluster of particles (or bodies) $Q_1, Q_2, Q_3$ and $Q_4$. The coefficients $M_n^m$ and $V_n^m$ are found using only the information on the locations and the strength (or mass) of the particles. No information on the

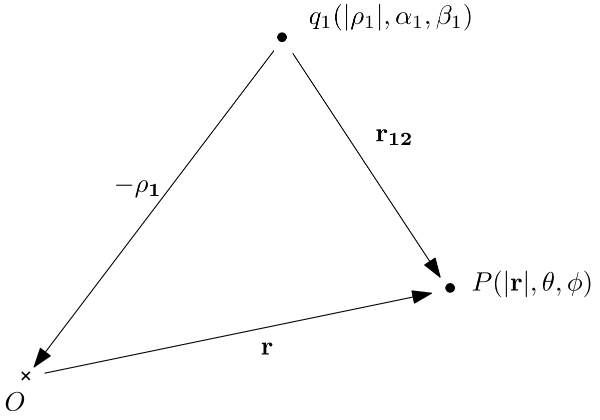
$q_1(|\rho_1|, \alpha_1, \beta_1)$
$\mathbf{r_{12}}$
$-\rho_\mathbf{1}$
$P(|\mathbf{r}|, \theta, \phi)$
$\mathbf{r}$
$O$

Fig. 6. Force acting on a point P. Note that $\mathbf{r_{12}} = \mathbf{r} - \rho_\mathbf{1}$

point of evaluation is required to compute the moments. These moments can be independently used for evaluating the force at a distant point $P$.

## C. A Tree Based Scheme for Force Computations

The treecode for the force computation is the same as the one used for computing the potentials of the form $r^{-\lambda}$ which was discussed in Chapter I. For the sake of completeness and in order to make this chapter stand-alone, we describe it again and give its pseudo-code.

The treecode can be viewed as a variant of either Barnes-Hut algorithm [4] or FMM [7] that uses only particle-cluster potential evaluations. The method works in two phases: the tree construction phase and the potential computation phase. In the tree construction phase, an oct-tree representation of the domain is computed. At each step of this phase, if the domain contains more than one particle, it is recursively divided into eight equal sub-domains. This process continues until each sub-domain has at most one particle. The resulting tree is an unstructured oct-tree.

Each internal node in the tree computes and stores multipole series representation of

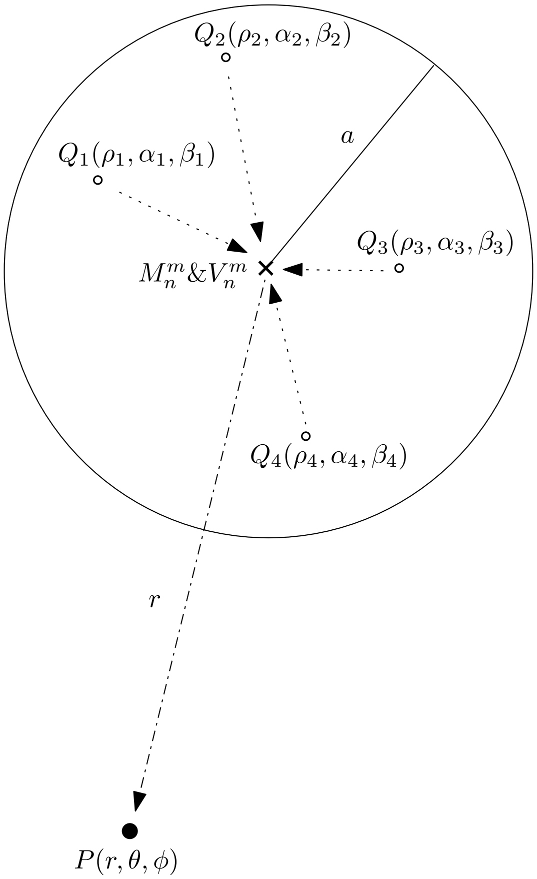
$Q_2(\rho_2, \alpha_2, \beta_2)$
$Q_1(\rho_1, \alpha_1, \beta_1)$
$a$
$Q_3(\rho_3, \alpha_3, \beta_3)$
$M_n^m \& V_n^m$
$Q_4(\rho_4, \alpha_4, \beta_4)$
$r$
$P(r, \theta, \phi)$

Fig. 7. Multipole expansion for forces

the particles within its sub-domain. Note that we don't have the means to compute translations of multipole coefficients from child to parent nodes. At each level of the tree we compute the multipole coefficients at every internal node directly from the particles contained in that node. These coefficients are obtained using (3.9). Once the tree has been constructed, the potential at a point is computed using the multipole coefficients of a subset of the nodes in the tree. The sub-domains of these nodes do not overlap, and cover the entire domain. Every such node must be sufficiently far away from the point. A specific constant $\alpha$ is used to check if a node is far enough from the evaluation point. Given the distance of a point from the center the sub-domain $d$, and the dimension of the sub-domain $r$, a point is considered to be far away from a node if the *multipole acceptance criterion* ($MAC$) defined by,

$$MAC = \frac{d}{r} \tag{3.10}$$

is greater than $\alpha$. The algorithm proceeds by applying the $MAC$ to the root of the tree to determine whether an interaction can be computed; if not, the node is expanded and the process is repeated for each of its eight children. If the node happens to be a leaf, then the potential is calculated directly by using the particles in the node. In order to improve the computational efficiency of the algorithm, the minimum number of particles in any leaf box can be set to a constant $s$. The pseudocode describing the algorithm is given at the end of this chapter in Algorithm 5.

### D. Numerical Experiments

We conduct two experiments. In these tests, a direct summation code is the benchmark for comparing errors and execution time. For star-cluster applications, Mcmillan and Aarseth [5] compares accuracies for a problem size of $10^3$. They target median force accuracy of $10^{-4}$ and achieve it using octupole moments for $MAC < 0.5$.

Table V and VI shows the error in the force $F$ incurred by truncating expression (3.8) after $p$ terms. The error $\delta F/F$ is defined by

$$\left(\frac{\delta F}{F}\right)^2 = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{|F_{tree} - F_{direct}|}{F_{scale}}\right)^2,$$

where $F_{direct}$ is the force determined by direct summation, $F_{tree}$ is the result returned by the tree algorithm and $F_{scale} = \sum_{ij} m_i/(r'_{ij})^2$ is a characteristic scale against which accuracy is measured. Here $m_i$'s are the masses and the sums are taken over all particles in the system. Table V compares $MAC$ with accuracy for the system size $N = 10^3$ in uniform random distribution and random positive masses between 0 and

1. The maximum number of particles in leaf boxes were fixed at 10. For $MAC = 0.9$ we see that an accuracy of $10^{-5}$ is reached using multipole degree 2. Also, as $MAC$ decreases, the error reduces proportionally. For a typical $MAC$ between 0.7 and 0.6 used in the Barnes-Hut algorithm, we have an order of magnitude reduction in the error for increasing multipole degrees. In [5], for $MAC < 0.3$ the accuracy of the octupole component of the force deteriorates markedly. In our case we see that dramatic reduction in the error is obtained as we increase the multipole degree. Also, Mcmillan et al. require $MAC < 0.5$ for accurate computations. Since the amount of direct computation is inversely proportional to $MAC$, it may be expensive to select such small $MAC$ for large systems.

Table VI and VII demonstrate the effect of $MAC$ on accuracy and execution time for 10000 particles in random uniform distribution. Each particle carries a random positive or a negative charge. This test is carried out to verify the numerical robustness and consistency of our approach for electrostatic force interactions. As before, the multipole degree was allowed to vary between 2 and 4 and the number of particles per leaf box was fixed at 10. Here we can see errors range between $10^{-6}$ and $10^{-7}$ and they are smaller compared to Table V. In Table VII we see a rapid increase in the execution time as we increase the multipole degree whereas the growth is smaller with

Table V. $MAC$ Vs Error for 1000 particles of positive masses

| | $MAC$ | | | | | |
|---|---|---|---|---|---|---|
| *Degree* | 0.9 | 0.8 | 0.7 | 0.6 | 0.5 | 0.4 |
| 2 | $2.8 \times 10^{-05}$ | $1.7 \times 10^{-05}$ | $1.0 \times 10^{-05}$ | $5.4 \times 10^{-06}$ | $2.6 \times 10^{-06}$ | $1.1 \times 10^{-06}$ |
| 3 | $2.2 \times 10^{-05}$ | $1.3 \times 10^{-05}$ | $8.1 \times 10^{-06}$ | $4.0 \times 10^{-06}$ | $1.4 \times 10^{-06}$ | $5.5 \times 10^{-07}$ |
| 4 | $1.0 \times 10^{-05}$ | $5.1 \times 10^{-06}$ | $2.3 \times 10^{-06}$ | $9.1 \times 10^{-07}$ | $3.0 \times 10^{-07}$ | $< 10^{-07}$ |

Table VI. *MAC* Vs Error for 10000 particles of positive and negative charges

| | *MAC* | | | | | |
|---|---|---|---|---|---|---|
| *Degree* | 0.9 | 0.8 | 0.7 | 0.6 | 0.5 | 0.4 |
| 2 | $7.0 \times 10^{-06}$ | $4.5 \times 10^{-06}$ | $2.8 \times 10^{-06}$ | $1.5 \times 10^{-06}$ | $7.7 \times 10^{-07}$ | $3.3 \times 10^{-07}$ |
| 3 | $4.3 \times 10^{-06}$ | $2.5 \times 10^{-06}$ | $1.3 \times 10^{-06}$ | $6.6 \times 10^{-07}$ | $2.7 \times 10^{-07}$ | $< 10^{-07}$ |
| 4 | $2.8 \times 10^{-06}$ | $1.3 \times 10^{-06}$ | $6.7 \times 10^{-07}$ | $2.8 \times 10^{-07}$ | $< 10^{-07}$ | $< 10^{-07}$ |

Table VII. *MAC* Vs Execution time (in secs) for 10000 particles

| | *MAC* | | | | | |
|---|---|---|---|---|---|---|
| *Degree* | 0.9 | 0.8 | 0.7 | 0.6 | 0.5 | 0.4 |
| 2 | 4.3 | 5.7 | 7.7 | 11.0 | 16.3 | 25.5 |
| 3 | 7.7 | 10.1 | 13.7 | 19.6 | 28.9 | 45.4 |
| 4 | 12.0 | 15.7 | 21.1 | 30.1 | 44.4 | 69.6 |

respect to the MAC. In order to improve the accuracy for a small system size, it is better to reduce the MAC than increasing the multipole degree. However, tree-based force evaluation schemes are quite inefficient for small number of particles because the overhead involved in manipulating and traversing the tree data structure is fairly large. For $N > 10^4$, the treecode presented in this chapter is expected to outperform direct summation algorithm in execution time.

**Algorithm 5** Force Calculation Algorithm for Particle Simulations

```
Require: Particles and their masses, multipole approximation degree, $p$, multipole
    acceptance criteria $\alpha$ constant that defines $MAC$ as shown in (3.10), maximum
    number of particles in a leaf box, $s$.
Ensure: Particles with the corresponding forces acting on it.
    // Construct oct-tree with leaf box having at most $s$ particles.
MULTIPOLE-CALCULATION ($p$)
for each level in the oct-tree do
    for each node in the level do
        use (3.9) to find the multipole coefficients with respect to the box centers.
    end for
end for
FORCE-COMPUTATION ()
for each particle do
    initialize node to root.
    if if $MAC(particle, node) > \alpha$ then
        use (3.8) to find the far field forces $F_{far}$.
    else if $node = leaf$ then
        compute near field forces $F_{near}$ directly.
    else
        apply $MAC$ to the children of the node recursively.
    end if
end for
Total Force $F = F_{near} + F_{far}$.
```

# CHAPTER IV

## A FAST TREECODE FOR PARTICLE SIMULATIONS

In this chapter, we propose a scheme in which we re-use the multipole moments computed during a time step, for future time increments. The basic idea here is the representation of the potential computed with respect to a new location $L_2$ in terms of the multipole moments at an old location $L_1$ and the dipole moments at the mid-point joining $L_1$ and $L_2$. We derive an analytic bound for the error in the potential when the moments computed at $L_1$ are re-used when the particle moves to $L_2$. We derive this result using a translated coordinate system in the mid-point joining $L_1$ and $L_2$. This asymptotic bound when analyzed in the context of an oct-tree, provides us with a treecode in which fewer updates to the multipole moments for the ancestor nodes are sufficient to compute the potential to a good accuracy. Naturally, the resulting treecode has a reduced execution time. The distinguishing feature of this algorithm is that the accuracy is not compromised in the need for reduced computational cost since the idea of computing fewer moments is based on the observation that the error bound in computing the potential rapidly decreases as node's spatial extent increases. This algorithm can be used in astrophysical or molecular dynamics $N$-body solvers where faster execution time per time step is a required to run large simulations. Finally, we provide numerical experiments that show the efficiency of the treecode.

The chapter is organized as follows: Section A discusses the multipole expansion theorems for $r^{-1}$ functions. In Section B we prove a theorem that shows the potential computed with respect to a new location $L_2$ can be represented using the potential at an old location $L_1$ and the dipole moments at the mid-point joining $L_1$ and $L_2$. We also derive an error bound for such an approximation. In Section C we discuss about the error bound. Section D describes an algorithm to compute the potentials using the multipole moments that are retained during time steps. Section E discusses the implementation issues.

## A. The First Addition Theorem for Multipole Expansions

This section describes the multipole expansion theorems from the classical FMM [7]. The reader is directed to Chapter II, Section A for a discussion about the following theorem. The proof of the First Addition Theorem for Multipole Expansions is given in the appendix. We note that the theorem given in the appendix slightly varies from the one given in this section. However, it is easy to see that both are equivalent. The following theorem is also given in Chapter II (see Theorem 1). For the sake of completeness, we again present it here.

**Theorem 6.** *Suppose that a charge of strength* $q$ *is located at* $Q(\rho, \alpha, \beta)$ *with* $\rho < a$. *At any point* $P(r, \omega, \phi)$ *with* $r > a$, *the potential* $\Phi(P)$ *is given by*

$$\Phi(P) = \frac{q}{r'} = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \frac{M_n^m}{r^{n+1}} Y_n^m(\omega, \phi), \tag{4.1}$$

*where* $M_n^m = q\rho^n Y_n^{-m}(\alpha, \beta)$.

When we truncate the expansion (4.1) at $n = p$, we have the error bound for the approximation as follows.

For any $p \geq 1$,

$$\left| \Phi(P) - \sum_{n=0}^{p} \sum_{m=-n}^{n} \frac{M_n^m}{r^{n+1}} Y_n^m(\omega, \phi) \right| \leq \left( \frac{q}{r - \rho} \right) \left( \frac{\rho}{r} \right)^{p+1}. \tag{4.2}$$

**Remark 1.** *If there are several charges* $\{q_i : i = 0, 1, \ldots k\}$ *around* $Q$ *with spherical coordinates* $\{(\rho_i, \alpha_i, \beta_i) : i = 1, 2, \ldots k\}$, *we can superpose them at* $Q$ *and the resulting multipole moment at* $Q$ *would be* $M_n^m = \sum_{i=0}^{k} q_i \rho_i^n Y_n^{-m}(\alpha_i, \beta_i)$.

We denote the moment vector

$$\mathbf{M} = \left[M_0^0, M_1^{-1}, M_1^0, M_1^1, \ldots, M_n^{-n}, \ldots, M_n^n, \ldots\right],$$

where $\left[M_0^0, M_1^{-1}, M_1^0, M_1^1, \ldots, M_n^{-n}, \ldots, M_n^n, \ldots\right]$ is a vector of multipole coefficients whose entries $\{M_i^j : i = 1, \ldots, \infty; 0 \leq |j| \leq i\}$ are defined in Theorem 6. Also, let $\mathbf{M}_p$ be the vector $\mathbf{M}$ truncated at $n = p$.

Suppose we have a vector of multipole moments $\mathbf{M}$ evaluated with respect to a point $Q$ around which the charges are located. We can then find the moments with respect to an arbitrary point $O$ by translating $\mathbf{M}$ to $O$. Formally,

**Theorem 7.** *Let* $\mathbf{M}$ *be a vector of multipole moments evaluated with respect to* $Q$ *around which charges of strengths* $q_1, \ldots q_N$ *are located within a distance of* $\xi$*. Also, let* $P$ *and* $Q$ *have coordinates* $(r, \omega, \phi)$ *and* $(\rho, \alpha, \beta)$*, respectively, with respect to an arbitrary point* $O$*. If* $r > \rho + \xi$*, the potential* $\Phi(P)$ *can be expanded as,*

$$\Phi(P) = \sum_{j=0}^{\infty} \sum_{k=-j}^{j} \frac{T_j^k(\mathbf{M})}{r^{j+1}} Y_j^k(\omega, \phi), \tag{4.3}$$

*where*

$$T_j^k(\mathbf{M}) = \sum_{n=0}^{j} \sum_{m=-n}^{n} \frac{M_{j-n}^{k-m} \cdot i^{|k|-|m|-|k-m|} \cdot A_n^m \cdot A_{j-n}^{k-m} \cdot \rho^n \cdot Y_n^{-m}(\alpha, \beta)}{A_j^k}, \tag{4.4}$$

*with* $A_n^m$ *defined by*

$$A_n^m = \frac{(-1)^n}{\sqrt{(n-m)! \cdot (n+m)!}}.$$

*Furthermore, for* $p \geq 1$

$$\left| \Phi(P) - \sum_{j=0}^{p} \sum_{k=-j}^{j} \frac{T_j^k(\mathbf{M})}{r^{j+1}} Y_j^k(\omega, \phi) \right| \leq \left( \frac{\sum_{i=1}^{N} q_i}{r - (\rho + \xi)} \right) \left( \frac{\rho + \xi}{r} \right)^{p+1}.$$

The reader is advised to refer [7] for a proof of these theorems. We note that (4.3) can be written as an inner product of two vectors as,

$$\Phi(P) = \langle \mathbf{T}_{\rho,\alpha,\beta}\mathbf{M}, \mathbf{R} \rangle,$$

where the $(r,s)^{th}$ entry of the matrix $\mathbf{T}_{\rho,\alpha,\beta} = (T_r^s)$ with $r = j^2+j+k$, $s = n^2+n+m$ and the indices as specified in (4.3) and (4.4), is defined by,

$$T_r^s = \frac{i^{|k|-|m|-|k-m|} \cdot A_{j-n}^{k-m} \cdot A_n^m \cdot \rho^{j-n} \cdot Y_{j-n}^{-(k-m)}(\alpha,\beta)}{A_j^k}, \tag{4.5}$$

and the $(j,k)^{th}$ entry of the evaluation point vector $\mathbf{R} = (R_j^k)$ is given by,

$$R_j^k = \frac{Y_j^k(\omega,\phi)}{r^{j+1}}. \tag{4.6}$$

## B. An Expansion Theorem for a Particle in Motion

In this section we derive a theorem that essentially states that when a particle moves from location $L_1$ to $L_2$, the potential computed with respect to $L_2$ can be represented using the multipole moments at $L_1$ and the dipole moments at the mid-point joining $L_1$ and $L_2$. We also give an analytic bound for the error in approximating the potential by truncating the dipole moments.

Consider a particle with a unit charge at location $L_1$ moving to a location $L_2$ during the simulation. Let $M$ be the mid-point joining $L_1$ and $L_2$ and $P$ be the evaluation point. Also, let $(\delta\rho, \delta\alpha, \delta\beta)$ and $(\delta\rho', \delta\alpha', \delta\beta')$ be the spherical coordinates of $L_1$ and $L_2$, respectively, with respect to the coordinate system at $O$ translated to $M$ (See Figure 8) and $(r_{mid}, \zeta, \eta)$ be the coordinates of $P$ in that system. We then have the following result.

**Theorem 8.** *Let $(\rho,\alpha,\beta)$ and $(\rho_1,\alpha_1,\beta_1)$ be the coordinates of $M$ and $L_1$, respectively, and $(r,\theta,\varrho)$ be the coordinate of $P$, all with respect to $O$. Then, for any $\alpha < 1$, the*

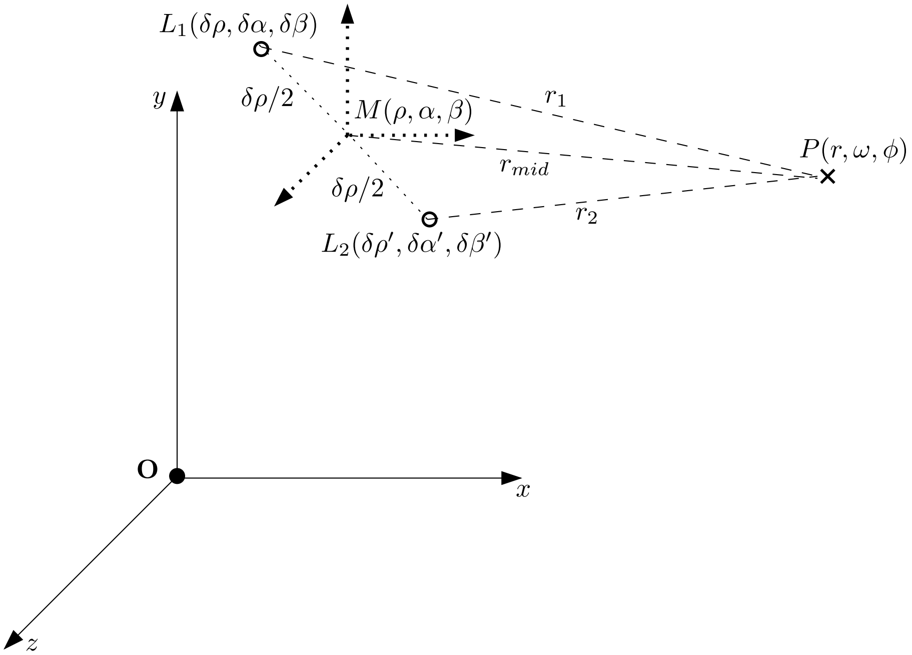

Fig. 8. Movement of a particle from location $L_1$ to $L_2$

*potential at* $P$ *with respect to* $L_2$, $\Phi_{L_2}(P)$, *when* $\frac{\rho_1}{r} < \alpha$ *and* $\frac{\delta\rho}{r_{mid}} < \alpha$, *is given by,*

$$\Phi_{L_2}(P) = \langle \mathbf{O}, \mathbf{R} \rangle + \langle \mathbf{D}, \mathbf{R_{mid}} \rangle \tag{4.7}$$

*where the moment* $\mathbf{O} = (O_j^k)$ *and the dipole moment* $\mathbf{D} = (D_j^k)$, *with* $j \geq 0$ *and* $0 \leq |k| \leq j$, *are defined as,*

$$O_j^k = \rho_1^j Y_j^{-k}(\alpha_1, \beta_1) \tag{4.8}$$

*and*

$$D_j^k = \begin{cases} -2(\delta\rho)^j Y_j^{-k}(\delta\alpha, \delta\beta) & \textit{if } j \textit{ is odd;} \\ 0 & \textit{if } j \textit{ is even.} \end{cases} \tag{4.9}$$

*Here* $\mathbf{R_{mid}}$ *and* $\mathbf{R}$ *are the evaluation point vectors with respect to* $M$ *and* $O$, *respectively, whose entries are defined in (4.6).*

*Furthermore, for any odd number* $p' \geq 1$, *we have the error for the approximation* *with truncated dipole moment* $\mathbf{D_{p'}}$ *(where* $j \leq p'$*),*

$$\hat{\Phi}_{L_2}(P) = \langle \mathbf{O}, \mathbf{R} \rangle + \langle \mathbf{D_{p'}}, \mathbf{R_{mid}} \rangle$$

*as*

$$|\Phi_{L_2}(P) - \hat{\Phi}_{L_2}(P)| \leq \frac{2}{r_{mid}} \cdot \frac{\left(\frac{\delta\rho}{r_{mid}}\right)^{p'+3}}{1 - \left(\frac{\delta\rho}{r_{mid}}\right)^2}. \tag{4.10}$$

*Proof.* Let $|L_1P| = r_1$, $|L_2P| = r_2$. Then we have,

$$\Phi_{L_2}(P) = \frac{1}{r_2} = \frac{1}{r_1} + \left(\frac{1}{r_2} - \frac{1}{r_1}\right)$$

Writing this in terms of multipole expansions,

$$\Phi_{L_2}(P) = \langle \mathbf{O}, \mathbf{R} \rangle + \sum_{j=0}^{\infty} \sum_{k=-j}^{j} \frac{(M_j'^k - M_j^k)}{r_{mid}^{j+1}} Y_j^k(\zeta, \eta), \tag{4.11}$$

where

$$\begin{aligned} M_j^k &= (\delta\rho)^j \, Y_j^{-k} \, (\delta\alpha, \delta\beta) \, , \\ M_j'^k &= (\delta\rho')^j \, Y_j^{-k} \, (\delta\alpha', \delta\beta') \, . \end{aligned} \tag{4.12}$$

Note that

$$\begin{aligned} \delta\rho' &= \delta\rho, \\ \delta\alpha' &= \pi - \delta\alpha, \\ \delta\beta' &= \begin{cases} \pi + \delta\beta & \text{if } \delta\beta < 0; \\ -\pi + \delta\beta & \text{if } \delta\beta > 0; \\ 0 & \text{if } \delta\beta = \delta\beta' = 0. \end{cases} \end{aligned}$$

Therefore for the associated Legendre functions and the exponentials we have,

$$P_j^{|k|}(\cos\delta\alpha') = \begin{cases} -P_j^{|k|}(\cos\delta\alpha) & \text{if } j+|k| \text{ is odd;} \\ P_j^{|k|}(\cos\delta\alpha) & \text{if } j+|k| \text{ is even,} \end{cases}$$

$$e^{-ik\delta\beta'} = \begin{cases} -e^{-ik\delta\beta} & \text{if } k \text{ is odd;} \\ e^{-ik\delta\beta} & \text{if } k \text{ is even.} \end{cases}$$

From (2.3), this implies,

$$Y_j^{-k}(\delta\alpha',\delta\beta') = \begin{cases} -Y_j^{-k}(\delta\alpha,\delta\beta) & \text{if } j \text{ is odd;} \\ Y_j^{-k}(\delta\alpha,\delta\beta) & \text{if } j \text{ is even.} \end{cases}$$

And so,

$$\begin{aligned} M_j'^k - M_j^k = D_j^k &= (\delta\rho)^j \left[Y_j^{-k}(\delta\alpha',\delta\beta') - Y_j^{-k}(\delta\alpha,\delta\beta)\right] \\ &= \begin{cases} -2(\delta\rho)^j Y_j^{-k}(\delta\alpha,\delta\beta) & \text{if } j \text{ is odd;} \\ 0 & \text{if } j \text{ is even.} \end{cases} \end{aligned} \tag{4.13}$$

Thus we have the result (4.7) from (4.11). We will now prove the error bound. Note that,

$$\Phi_{L_2}(P) = \sum_{j=0}^{\infty}\sum_{k=-j}^{j} \frac{M_j'^k}{r_{mid}^{j+1}} Y_j^k(\zeta,\eta)$$

and

$$\langle \mathbf{O}, \mathbf{R} \rangle = \sum_{j=0}^{\infty}\sum_{k=-j}^{j} \frac{O_j^k}{r^{j+1}} Y_j^k(\theta,\varrho) = \sum_{j=0}^{\infty}\sum_{k=-j}^{j} \frac{M_j^k}{r_{mid}^{j+1}} Y_j^k(\zeta,\eta)$$

and therefore,

$$\begin{aligned}|\Phi_{L_2}(P) - \hat{\Phi}_{L_2}(P)| &= \left|\sum_{j=0}^{\infty}\sum_{k=-j}^{j}\frac{(M_j'^k - M_j^k)}{r_{mid}^{j+1}}Y_j^k(\zeta,\eta) - \sum_{j=0}^{p'}\sum_{k=-j}^{j}\frac{D_j^k}{r_{mid}^{j+1}}Y_j^k(\zeta,\eta)\right|,\\ &= \left|\sum_{j=0}^{\infty}\sum_{k=-j}^{j}\frac{D_j^k}{r_{mid}^{j+1}}Y_j^k(\zeta,\eta) - \sum_{j=0}^{p'}\sum_{k=-j}^{j}\frac{D_j^k}{r_{mid}^{j+1}}Y_j^k(\zeta,\eta)\right|,\\ &= \left|\sum_{j=p'+1}^{\infty}\sum_{k=-j}^{j}\frac{D_j^k}{r_{mid}^{j+1}}Y_j^k(\zeta,\eta)\right|. \qquad (4.14)\end{aligned}$$

We know from (4.13) that, if $j$ is even, then $D_j^k = 0$ and when $j$ is odd (4.14) is same as,

$$|\Phi_{L_2}(P) - \hat{\Phi}_{L_2}(P)| \leq \left|\sum_{n=\lceil\frac{p'}{2}\rceil}^{\infty}\sum_{m=-(2n+1)}^{2n+1}\frac{2(\delta\rho)^{2n+1}Y_{2n+1}^{-m}(\delta\alpha,\delta\beta)}{r_{mid}^{2n+2}}Y_{2n+1}^{m}(\zeta,\eta)\right|,$$

in which case we have replaced $j = 2n+1$. Also, we have the inequality,

$$\left|\sum_{m=-(2n+1)}^{2n+1}Y_{2n+1}^{-m}(\delta\alpha,\delta\beta)Y_{2n+1}^{m}(\zeta,\eta)\right| = |P_{2n+1}(\cos\chi)| \leq 1,$$

where $\chi$ is the angle between $L_1MP$ and $P_{2n+1}(\cos\chi)$ is the Legendre polynomial of degree $2n+1$ (for instance, see [17] for the above inequality). Thus, when $p'$ is odd,

$$\begin{aligned}|\Phi_{L_2}(P) - \hat{\Phi}_{L_2}(P)| &\leq \frac{2}{r_{mid}}\sum_{n=\lceil\frac{p'}{2}\rceil}^{\infty}\left(\frac{\delta\rho}{r_{mid}}\right)^{2n+1}\\ &= \frac{2\cdot\delta\rho}{r_{mid}^2}\left[\sum_{n=0}^{\infty}\left(\frac{\delta\rho}{r_{mid}}\right)^{2n} - \sum_{n=0}^{\lfloor\frac{p'}{2}\rfloor}\left(\frac{\delta\rho}{r_{mid}}\right)^{2n}\right],\\ &= \frac{2\cdot\delta\rho}{r_{mid}^2}\left[\frac{1}{1-\left(\frac{\delta\rho}{r_{mid}}\right)^2} - \frac{1-\left(\frac{\delta\rho}{r_{mid}}\right)^{p'+2}}{1-\left(\frac{\delta\rho}{r_{mid}}\right)^2}\right],\\ &= \frac{2}{r_{mid}}\cdot\frac{\left(\frac{\delta\rho}{r_{mid}}\right)^{p'+3}}{1-\left(\frac{\delta\rho}{r_{mid}}\right)^2}.\end{aligned}$$

□

This theorem immediately gives us the following corollary for the total error bound for the potential approximation at $L_2$.

**Corollary 9.** *Let* $\mathbf{O_p}$ *and* $\mathbf{D_{p'}}$ *be the moments obtained by truncating the moments* $\mathbf{O}$ *and* $\mathbf{D}$ *defined in (4.8) and (4.9) at* $j = p$ *and* $j = p'$*, respectively. Also, let* $p' \leq p$*, be an odd positive integer. If* $\frac{\delta\rho}{r_{mid}} \leq \frac{1}{\sqrt{3}}$*, the total error bound for the approximate potential at* $L_2$*,*

$$\hat{\hat{\Phi}}_{L_2}(P) = \langle \mathbf{O_p}, \mathbf{R} \rangle + \langle \mathbf{D_{p'}}, \mathbf{R_{mid}} \rangle, \tag{4.15}$$

*is given by,*

$$|\Phi_{L_2}(P) - \hat{\hat{\Phi}}_{L_2}(P)| \leq E_1 + E_2 \tag{4.16}$$

*where*

$$E_1 = \frac{1}{r - (\rho + \frac{\delta\rho}{2})} \left( \frac{\rho}{r} + \frac{\delta\rho}{2r} \right)^{p+1},$$

*and*

$$E_2 = \frac{1}{r - \rho} \left( \frac{\delta\rho}{r - \rho} \right)^{p'+1}.$$

*Proof.* If $\frac{\delta\rho}{r_{mid}} \leq \frac{1}{\sqrt{3}}$, then,

$$1 - \left( \frac{\delta\rho}{r_{mid}} \right)^2 \geq 2 \left( \frac{\delta\rho}{r_{mid}} \right)^2.$$

Consequently, the error bound (4.10) satisfies,

$$|\Phi_{L_2}(P) - \hat{\Phi}_{L_2}(P)| \leq \frac{1}{r_{mid}} \cdot \left( \frac{\delta\rho}{r_{mid}} \right)^{p'+1}.$$

Also, for the truncated multipole expansions, we have the error bound defined by (4.2). i.e,

$$\left| \sum_{j=0}^{\infty} \sum_{k=-j}^{j} \frac{O_j^k}{r^{j+1}} Y_j^k(\theta, \varrho) - \sum_{j=0}^{p} \sum_{k=-j}^{j} \frac{O_j^k}{r^{j+1}} Y_j^k(\theta, \varrho) \right| \leq \left( \frac{1}{r - \rho_1} \right) \left( \frac{\rho_1}{r} \right)^{p+1}.$$

Thus, the bound in (4.16) can now be readily obtained by using the above two bounds in the triangle inequality,

$$|\Phi_{L_2}(P) - \hat{\hat{\Phi}}_{L_2}(P)| \leq |\Phi_{L_2}(P) - \hat{\Phi}_{L_2}(P)| + |\hat{\Phi}_{L_2}(P) - \hat{\hat{\Phi}}_{L_2}(P)|$$

and noting that $\rho_1 \leq \rho + \frac{\delta\rho}{2}$ and $r - \rho \leq r_{mid}$. □

### C. Discussion

The corollary essentially means that, when $\delta\rho \ll r - \rho$, the bound (4.16) is dominated by the first term than the second term, i.e, $E_1$ much larger than $E_2$. In particular, if

$$\frac{\delta\rho}{r} < \frac{\rho}{r}\left(1 - \frac{\rho}{r}\right)$$

the error bound is dominated by $E_1$. In Barnes-Hut type treecodes, the criteria for approximation satisfies $\frac{\rho}{r} < \alpha$, which is usually assigned a value of 0.7. Choosing the negative of this value as a lower bound for $-\frac{\rho}{r}$, we have

$$\frac{\delta\rho}{r} < 0.3 \cdot \left(\frac{\rho}{r}\right)$$

and since $\frac{\rho}{r}$ can be as large as 0.7, $\frac{\delta\rho}{r}$ could attain a maximum value of 0.21 while $E_2$ remains lesser than $E_1$. Further, we infer an important point from (4.16) - that for a given $p'$, $E_2$ approaches zero rapidly as $r$ increases. This when viewed in the context of an oct-tree means the following: when the movement of the particle is small, the error obtained by the approximation (4.15) is dominated by truncating the moment $\mathbf{O}$ as the nodes get closer to the root (as against truncating the dipole moment $\mathbf{D}$). We will illustrate this point in detail in the *Multipole Updating Phase* in the next section.

We note that in the approximation (4.15), the use of the dipole moment is considered at $M$ whereas the moment $\mathbf{O}$ is used with respect to $O$. We could equally consider the approximation using the dipole moment translated to $O$, namely, $\mathbf{T}_{\rho,\alpha,\beta}\mathbf{D}_{\mathbf{p'}}$, where the entries of $\mathbf{T}_{\rho,\alpha,\beta}$ are defined in (4.5). The difference should be small, since $\langle \mathbf{T}_{\rho,\alpha,\beta}\mathbf{D}_{\mathbf{p'}}, \mathbf{R}\rangle$ and $\langle \mathbf{D}_{\mathbf{p'}}, \mathbf{R}_{\mathbf{mid}}\rangle$ converge to the same limit as $p'$ approaches infinity. Similarly, instead of using the moment $\mathbf{O}$, we can translate the moment $\mathbf{M} = (M_j^k)$ given in (4.12) to $O$ for approximating the potential, i.e., use $\mathbf{T}_{\rho,\alpha,\beta}\mathbf{M}_{\mathbf{p}}$ in the place of $\mathbf{O}_{\mathbf{p}}$, where $\mathbf{M}_{\mathbf{p}}$ is the truncated moment $\mathbf{M}$ with $j \leq p$. Note that if the moments $\mathbf{M}$ and $\mathbf{D}$ are both truncated at $j = p'$, the translated moment $\mathbf{T}_{\rho,\alpha,\beta}(\mathbf{M}_{\mathbf{p'}} + \mathbf{D}_{\mathbf{p'}})$ is same as the multipole moment for the particle at location $L_2$ computed with respect to $M$ and then translated to $O$ (without the use of any dipole moment). Thus, we can think of the vector $\mathbf{T}_{\rho,\alpha,\beta}(\mathbf{M}_{\mathbf{p}} + \mathbf{D}_{\mathbf{p'}})$ as just a change in the first $(p'+1)^2$ coefficients of the moment $\mathbf{T}_{\rho,\alpha,\beta}\mathbf{M}_{\mathbf{p}}$, obtained by directly computing the multipole coefficients for the particle at $L_2$ with respect to $M$ and translating it to $O$ without using any dipole moment. Moreover, computing the translation can be avoided by computing $(p'+1)^2$ coefficients with respect to $O$ instead of $M$.

Finally, we also note that the analysis can be extended to several particles with varying charge strengths just by superimposing the moments. In this case, the error bound (4.16) will have a scaling factor which will have the sum of the magnitudes of the charge strengths.

## D. A Fast Treecode for Particle Simulations

The error bound derived in Section B can be applied to a treecode to accelerate the time to compute the potentials. Since the bound given in (4.16) shows that as the distance to the evaluation point increases the error bound is dominated $E_1$ than $E_2$, recomputing the entire multipole moments for all the nodes in the tree can be carried out only when the particles move sufficient distances. Otherwise, fewer moment updates can be made for the nodes in the tree depending on the height of the node. We propose one such algorithm in this section. This algorithm can be viewed as a variant of the Barnes-Hut treecode [4] or FMM [7] that uses only particle-cluster potential evaluations. An important aspect of this treecode is that it can be used in the existing implementations of FMM or the Barnes-Hut algorithm with little modification. In general, for a given time step, the method works in three phases: the tree construction phase, the multipole updating phase and the potential computation phase, however, the tree is reconstructed only on certain time steps.

*Tree Construction Phase:* In this phase, a spatial tree representation of the domain is derived as given in Chapter I Section 3 except that the decomposition of the space is continued until each sub-domain has at most $s$ particles instead of one particle. To determine the time step during which the tree needs to be reconstructed, we use the formula suggested in [25], i.e., if $n$ is the number density of the system and $l$ is the length of a leaf cell $L_c$ containing $n_L$ particles, the ratio between the time interval when the tree is reconstructed, $\Delta t_{tree}$, and the step size of the time integration, $\Delta t$, should obey,

$$\begin{aligned}\frac{\Delta t_{tree}}{\Delta t} &= \frac{l}{n^{-1/3}}, \\ &\simeq n_L^{1/3},\end{aligned}$$

since $l^3 n = n_L$. Therefore, we can consider reconstructing the tree based on the average number of particles present in the leaf nodes. Thus, if there are $T_L$ number of leaf nodes, the ratio between the tree reconstruction and interval for the time integration is given by,

$$\frac{\Delta t_{tree}}{\Delta t} = \left[\frac{1}{T_L}\sum_{L_c} n_L\right]^{1/3}. \tag{4.17}$$

Hence, in our algorithm, at every time step when the tree is reconstructed, we determine the next reconstruction time step based on (4.17). Also, during the tree construction, for each node, we store the *level*, $L$, which is defined to be the difference between the height of the tree and the depth of the node from the root. We note that if $n$ is the number density of the system, then $n^{-1/3}$ gives the average interparticle distance and so if the particles cover large distances during time steps, the tree construction has to be carried out often.

*Potential Computation Phase:* This is done exactly as in the Barnes-Hut scheme discussed in Chapter II Section C (by applying the multipole acceptance criterion to the node or each of its eight children, starting from the root). This phase is performed at each time step of the simulation.

*Multipole Updating Phase:* During the time steps the tree is reconstructed, the truncated multipole moments $\mathbf{O}_p$ are computed and stored for all the nodes in the tree directly from the particles with respect to the geometric center of the node. Note that we can equally compute the moments for the parent nodes from the child nodes using the translation operator. An essential aspect of this algorithm is re-using some of the coefficients of the moments $\mathbf{O}_p$ during the time steps the tree is retained. The key idea in doing this is as follows:

Consider a particle $Q$ in a cell $C$ and a sphere $S_C$ circumscribing $C$ such that the multipole expansion of $C$ is valid only outside $S_C$. Let $\nu$ be the radius of $S_C$ (see Fig. 9). Also, for the given particle in the cell $C$, let $F_C(\rho) = \frac{E_2(\rho)}{E_1(\rho)}$ denote the ratio of the errors $E_2$ and $E_1$ defined in (4.16) treated as a function of $\rho$. If $p = p'$, $\frac{\rho}{r} = \alpha < 1$

and $\delta\rho \ll r$, then,

$$\begin{aligned}F_C(\rho) = \frac{E_2(\rho)}{E_1(\rho)} &= \frac{1}{r-\rho}\left(\frac{\delta\rho}{r-\rho}\right)^{p+1} \cdot \left(r-\rho-\frac{\delta\rho}{2}\right)\left(\frac{2r}{2\rho+\delta\rho}\right)^{p+1}, \\ &\approx \frac{1}{r-\rho}\left(\frac{\delta\rho}{r-\rho}\right)^{p+1} \cdot (r-\rho)\left(\frac{r}{\rho}\right)^{p+1}, \\ &= \left(\frac{1}{\rho}\right)^{p+1} \cdot \left(\frac{\delta\rho}{1-\alpha}\right)^{p+1}.\end{aligned}$$

Since $F_C(\rho)$ compares the error obtained using the dipole moment with the error in using the standard multipole moment, it can be construed as a relative error bound for $Q$ in the cell $C$ having a displacement of $\delta\rho$. Note that for a given $\alpha$ and $\delta\rho$, $F_C(\rho)$ remains constant on a sphere of radius $\rho$. Therefore, for $p > 2$, the average relative error bound for $C$, $AVG_C$, can be given by,

$$\begin{aligned}AVG_C &= \frac{3}{4\pi\nu^3}\int_{\delta\rho}^{\nu} 4\pi\rho^2 \cdot F_C(\rho)d\rho, \\ &= \frac{3}{p-2}\cdot\left(\frac{\delta\rho}{1-\alpha}\right)^{p+1}\cdot\frac{1}{\nu^{p+1}}\left[\left(\frac{\nu}{\delta\rho}\right)^{p-2}-1\right], \\ &= O\left(\frac{1}{\nu^{p+1}}\left[\frac{\nu}{\delta\rho}\right]^{p-2}\right), \qquad (4.18)\end{aligned}$$

assuming $\nu \gg \delta\rho$. Let us now consider an ancestor cell $A$ of the cell $C$ which contains $Q$. Let $\gamma$ be the radius of the sphere $S_A$ circumscribing $A$ such that the multipole expansion of $A$ is valid only outside the sphere $S_A$ (see Fig. 9). Further, let $C$ be the $d^{th}$ descendent of $A$. Since $2^d\nu = \gamma$, from (4.18) we have,

$$AVG_A = O\left(\frac{1}{\gamma^{p+1}}\left[\frac{\gamma}{\delta\rho}\right]^{p-2}\right) = \frac{1}{8^d}\cdot O\left(\frac{1}{\nu^{p+1}}\left[\frac{\nu}{\delta\rho}\right]^{p-2}\right) = \frac{1}{8^d}\cdot AVG_C.$$

Therefore, the average relative error for an ancestor cell is much lesser than the descendent cell when the particle moves a small distance. In other words, in general, if the relative error bound $F_C(\rho)$ is small for a particle in a cell $C$, it should be much

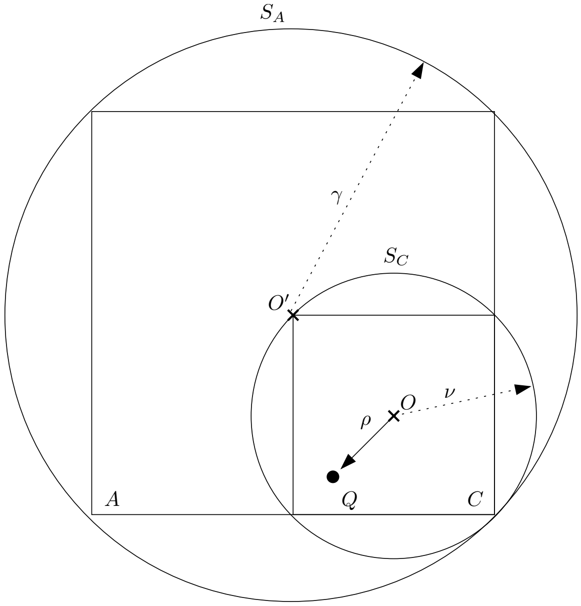

Fig. 9. Cell $C$ and its ancestor cell $A$ containing a particle $Q$.

smaller when considered with respect to an ancestor cell $A$ of $C$. Even though the total error depends on the evaluation point and the particle location, it is reasonable to expect that the cumulative errors for the ancestor nodes to be lesser than the errors for the descendent nodes for the particles near the center of the cells, when the multipole acceptance criteria is satisfied. This observation along with the approximation given in (4.15) means that fewer dipole moment coefficients are required for the ancestor nodes than the descendent nodes during the time steps the tree is retained. That is, $p'$ can be decreased for the ancestor nodes as the dipole moments $\mathbf{D}_{p'}$ are computed for the particle in the new location. From the discussion in Section C following the error bound, computing the dipole moments $\mathbf{D}_{p'}$ is same as replacing the $(p'+1)^2$ coefficients of $\mathbf{O}_p$ with the moments computed at the new location with respect to the center of the cell. Thus, in our algorithm, we reduce $p'$ by a factor of two for the parent compared to its children. Hence, at any time step $t$ after the tree construction phase, the moment for a node at level $L$ is computed by replacing the first $(p'+1)^2$ coefficients of the multipole moment $\mathbf{O}_p$ with the new moment $\mathbf{O}^t_{p'}$ where $p' = \lfloor p/2^{L+1} \rfloor$ and $\mathbf{O}^t_{p'}$ is the moment computed directly from the particles. Figure 10 illustrates the multipole updating phase for a particle where the multipole degree $p$ is fixed at 4. This illustration assumes the simulation is run for several time steps $t_i$, $i = 0, 1, \ldots$ and $t_{tree}$ denotes the time at which the tree is reconstructed. The particle $P$ is in the leaf box $L$ having center $C_1$ and has three ancestor boxes $A_1, A_2$ and $A_3$ with centers $C_2, C_3$ and $O$, respectively. The multipole vectors with respect to the centers are denoted by $v_1, v_2$ and $v_3$. The ancestor box $A_3$ (or the root node) does not have a multipole vector because it cannot be used to compute potentials as all the particles

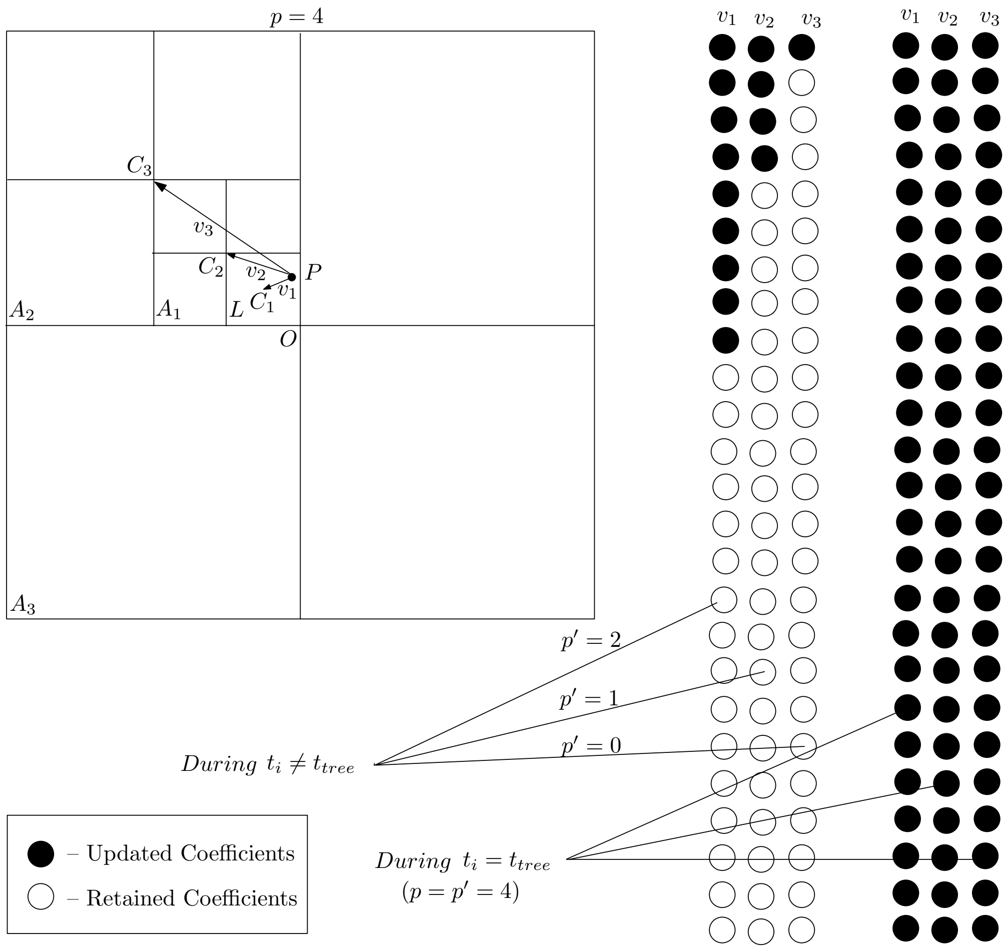

Fig. 10. Illustration of multipole updating phase for a particle with $p = 4$.

lie within it. Since $p = 4$, we have 25 multipole coefficients in the moment vectors. If $t_i = t_{tree}$ then all the coefficients of the moment vectors have to be updated. When $t_i \neq t_{tree}$, we have $p' = 2, 1$ and $0$ for the boxes $L, A_1$ and $A_2$, respectively. And therefore $9, 4$ and $1$ coefficient(s) are updated correspondingly in the existing vectors $v_1, v_2$ and $v_3$.

The pseudocode for the algorithm is given at the last page of this chapter in Algorithm 6. We have implemented the treecode in a simulation. The details of the numerical experiments and the results are discussed in Section E.

## E. Numerical Experiments

A simulation that uses the proposed treecode was implemented in C++ programming language and the computations were performed in double precision on a 2.4 GHz, 512 MB Intel P4 PC running Fedora 2.0. Standard Template Library (STL) data structures were used for efficient memory and time management for the dynamically adaptive tree construction phase. The constants that appear in spherical harmonics and Associated Legendre polynomials such as factorials and normalization factors were precomputed and used to reduce the overall computation time. The simulation is carried out for several time steps $t_i, i = 0, 1, \ldots, k$ ($k$ being a parameter) during which the potentials are evaluated at every step. Initially at $t_0$, the code first constructs the oct-tree and computes the multipole moments for all the nodes in the tree. After this initial tree construction phase, at every time step $t_i$, a random displacement $\delta\rho_i$ is introduced for all the particles in the leaf nodes. This displacement is fixed at one half of the radius of the farthest leaf node (based on [5]). At any particular tree construction step, the subsequent time step at which the tree should be reconstructed is determined using the formula (4.17). For all the time steps during which the tree is preserved, some of the coefficients of the multipole moments that are determined at the nearest tree construction time step, are replaced (if required) by the new moments that are computed. (as described in *Multipole Updating Phase* in Section D).

We conduct three experiments to study the behavior of our treecode $T$. In all the experiments, our treecode is compared with a standard treecode $S$ in which the multipole updates are carried out for all the nodes at every time step. Typically, in a treecode, translation operators are used to update the multipoles for the parent nodes while they are directly calculated for the leaf node particles. However, we have avoided the use of these operators as they produce additional errors in evaluating the potential. Thus, in our standard treecode $S$, the moments are updated for all the nodes using the particles contained in them. We also use a direct summation routine as a benchmark for comparing the errors. In these experiments, a random uniform particle distribution on a cube of length 1000 is used. All the particles carry a unit charge. The maximum number of particles allowed in a leaf box is fixed at one hundredth of the total number of particles. A random run in between the tree reconstruction time step is chosen and the results are detailed.

The parameters in the experiments are: multipole acceptance criteria constant, $\alpha$, total number of particles in the system, $N$, multipole truncation degree during the tree construction time, $p$, and the multipole degree used during every time step when the tree is retained, $p'$ . In the experiments, we study the errors resulting from the treecodes $T$ and $S$ by varying these parameters. To measure the error, we use the maximum percentage relative error (PRE) for a treecode $C$ defined by

$$PRE_C = 100 * \max_i \left[ \frac{|\Phi_{tree}(i) - \Phi_{direct}(i)|}{|\Phi_{direct}(i)|} \right],$$

where $\Phi_{tree}(i)$ is the potential for the $i^{th}$ particle obtained using the treecode and $\Phi_{direct}(i)$ is the potential of the $i^{th}$ particle computed using the direct summation algorithm. The cost for computing the multipole moments (in seconds) are also reported for these experiments.

Table VIII shows the errors and the time spent in updating the multipole moments for different choices of the $MAC$ constant $\alpha$. Note that $\alpha$ determines when an interaction can be computed and thus it is directly related to the error in the treecode. For this experiment, the values of $\alpha$ were fixed at 0.90, 0.80 and 0.70 and the multipole degree during the tree construction time was chosen to be 3. The number of particles $N$ was determined at 8192 for this experiment. The average height of the tree was 4 and the trees were constructed approximately once in every four time steps. We observe that the accuracy is more or less the same in our treecode although we have only updated very few nodes as opposed to the treecode $S$ in which all the nodes were updated. Since we reduce the multipole truncation degree for the nodes by a factor of two for a unit increase in the node level, only the farthest leaf nodes were updated (nodes with level 0) with new moments. These moments were truncated with multipole degree $p' = \lfloor 3/2 \rfloor = 1$. Consequently, there is a reduction in the computation time. The savings in the execution time would increase as we increase $\alpha$ because the direct interaction time increases for the nodes that doesn't satisfy the multipole acceptance criteria at any level. In general, astrophysical simulations require around 1% accuracy and this result indicates that one may obtain at least $40 - 50\%$ savings per time step in the multipole computation phase. We note that the time reported here is for computing the multipole moments which doesn't reduce with $\alpha$.

In Table IX, we vary the number of particles $N$ and report the time for the multipole updates and the errors for a specific value of $\alpha$ and $p$. The $MAC$ constant $\alpha$ in the system is fixed at 0.90 and the multipole degree $p$ is 4. $N$ ranges from 512 to 32768. The average height of the tree $H$ is also shown. In Table X, the time and

Table VIII. Error and the time for multipole updates for the treecodes varying $\alpha$

| | $N = 8192$ | | | |
|---|---|---|---|---|
| $\alpha$ | $PRE_T$ | $PRE_S$ | $\text{Time}_T$ | $\text{Time}_S$ |
| 0.90 | 1.273 | 0.898 | 0.83 | 1.65 |
| 0.80 | 0.785 | 0.362 | 0.83 | 1.65 |
| 0.70 | 0.209 | 0.094 | 0.83 | 1.65 |

the errors are given as a function of $p$. Here, $N$ is fixed at 8192 and $\alpha = 0.90$.

Table IX. Error and the time for multipole updates for treecodes varying $N$

| | $\alpha = 0.90, p = 4$ | | | | |
|---|---|---|---|---|---|
| $N$ | $H$ | $PRE_T$ | $PRE_S$ | $\text{Time}_T$ | $\text{Time}_S$ |
| 512 | 3 | 0.463 | 0.213 | 0.59 | 0.81 |
| 1024 | 4 | 0.692 | 0.431 | 0.76 | 1.13 |
| 4098 | 4 | 0.651 | 0.692 | 0.94 | 1.42 |
| 8192 | 4 | 0.834 | 0.643 | 1.23 | 1.89 |
| 32768 | 6 | 0.837 | 0.656 | 1.84 | 4.62 |

From Table IX, it can be observed that for a reasonable accuracy of 1%, the time for performing the multipole updates in our treecode is almost 33% lesser than the standard treecode $S$ when the average tree height is 4. When the average tree height is 3, the time difference is not much and we have more than 50% savings when the average height of the tree has increased to 6. Notice that the number of levels that require updates depends on the multipole degree $p$ and in our case the updates are done for level 0 and 1. Thus, when $H$ is 3, only the children of the root did not have any updates and so the savings is less. For $H = 6$, we have four levels

Table X. Error and the time for multipole updates for treecodes varying $p$

| | $N = 8192, \alpha = 0.90$ | | | |
|---|---|---|---|---|
| $p$ | $PRE_T$ | $PRE_S$ | $\text{Time}_T$ | $\text{Time}_S$ |
| 4 | 0.873 | 0.598 | 1.19 | 1.83 |
| 5 | 0.544 | 0.145 | 2.13 | 3.52 |
| 6 | 0.115 | 0.062 | 3.34 | 5.35 |
| 7 | 0.093 | 0.011 | 5.12 | 8.27 |

that did not have any fresh computation of the multipoles during the time the tree is preserved. Since, computing only a few moments has a considerable advantage, the savings in time for $N = 32768$ is large. In Table X, it can be seen that with the increase in the multipole degree $p$, we have a large improvement in the time during the multipole computation phase. And the accuracy almost remains the same in both codes. Note that the time to compute multipole moments of degree $p$ is proportional to $p^2$. And thus we have a good savings in the time in our code. This is especially useful for molecular dynamics simulations where accuracy cannot be compromised in the need for faster evaluation of the potentials and could be equally applied in the simulation of star clusters where the accuracy requirement is presently limited by octopole moments [6].

**Algorithm 6** Treecode for Fast Potential Approximations

```
Require: Particles and their charge strengths, multipole approximation degree, $p$,
    multipole acceptance criteria, $\alpha$, and maximum number of time steps, $k$.
Ensure: Potential acting on each particle during each time step $t_i$, $i = 0, \dots, k$.
    // $t_{tree}$ is the tree construction time step. Initially, $t_{tree} = t_0$.
    // If $t_{tree} > t_k$, we set $t_{tree} = t_k$

for $i = 0, 1, \dots, k$ do
  if $t_i = t_{tree}$ then
    Construct oct-tree T.
    for each node $N$ in $T$ do
      Compute the level of of the node $N$, $N_l = \text{Height}(T) - \text{Depth}(N)$.
      Compute the multipole vector of degree $p$ with respect to the center of $N$.
    end for
    Use (4.17) to calculate the next tree construction time step, $t_{tree} = t_{tree} + \lceil \frac{\Delta t_{tree}}{\Delta t} \rceil$.
  else
    for each node $N$ in $T$ do
      Compute the multipole vector of degree $p' = \lfloor p/2^{N_l+1} \rfloor$ using the new locations.
      Update $(p'+1)^2$ coefficients of the existing multipole vector with the computed moment vector.
    end for
  end if
  Evaluate the potential on each particle using $\alpha$ as described in Section C.
  Update the change in the location for each particle by computing the forces.
end for
```

# CHAPTER V

# CONCLUSION AND FUTURE DIRECTIONS

## A. Conclusion

This dissertation was aimed at answering some important questions that arise in $N$-body simulations, particularly in potential approximation and force evaluation of individual particles. We saw in Chapter I about the limitations of the current methods when it comes to efficiently evaluating the potentials of the form $r^{-\lambda}$ or when the particles are in motion. In Chapters II, III and IV, we proposed treecodes that addressed these limitations and we provided a different representation for the potentials and forces based on the relationship between Ultraspherical and Legendre polynomials. Specifically, in Chapter II, a new multipole expansion based treecode for computing the potentials of the form $r^{-\lambda}$ was introduced and this expansion theorem was used to represent gravitational forces in Chapter III. We also compared our method with the existing schemes and showed that our treecode was asymptotically superior than the compared methods. Moreover, the force representation in Chapter III was used in a treecode to compute accurate force interactions between the particles. Further, the advantage of our treecode was illustrated by surveying some of the methods that are used for accurate description of the forces. In Chapter IV, we presented an efficient treecode in which we updated the node's moments based on the particle's movement. This was based on an asymptotic bound for the error in the potential when particles move in the system. This bound provided us the reason for retaining the oct-tree and performing fewer multipole updates to the parent nodes whose centers are far away from the leaf node centers. A remarkable characteristic of this treecode was that the accuracy was not compromised in the need for reduced computational cost. Further, we demonstrated the efficiency of our treecodes using several numerical experiments.

## B. Future Directions

There are several ways to extend the present work. For example, the multipole expansion theorem for $r^{-\lambda}$ potentials introduced in Chapter II would allow addition theorems that could translate the moments from one location to another. The Fast Multipole Method uses these translations to reduce the complexity from $O(N \log N)$ to $O(N)$ for the Coulomb potential. A similar strategy could be devised for $r^{-\lambda}$ potentials using spherical coordinates. Of course, these addition theorems can be devised for the multipole expansion theorem for the forces described in Chapter III. Although the performance of the treecodes in Chapter II and Chapter III were impressive, it would be interesting to see the results of these codes when augmented with a symplectic integrator in a real simulations. Furthermore, the execution of these codes on various distributions such as the Plummer model, which is widely used in astrophysics, could be studied. The scope for further research based on the results in Chapter IV is enormous. For instance, when a particle exceeds the leaf box dimension one can increase the multipole acceptance criteria and still maintain the same accuracy without re-computing the multipole coefficients for all the nodes in the tree. This can be accomplished using the asymptotic bound we derived. Such a method will be really useful since computing the moments for all the nodes in the tree is an expensive operation. However, increasing the multipole acceptance criteria will increase the execution time as well. An analysis about the tradeoff will be worth pursuing. Another interesting application of our bound in Chapter IV will be in the simulations that use the translation operators. Typically, the multipole moments are translated between the child and the parent nodes. We have avoided the use of such operators as they result in increased errors. Even though, such operators produce additional errors, they reduce the computational cost to a large extant and in general, astrophysical simulations tolerate much larger errors (around 1%) than we have considered. Accordingly, the use of our bound in conjunction with the schemes that use the translations can be analyzed. More importantly, it will be highly beneficial to examine the performance of our method in astrophysical simulations, where Barnes-Hut and its variants are highly popular.

# REFERENCES

[1] K. Srinivasan, H. Mahawar, V. Sarin, A Multipole Based Treecode Using Spherical Harmonics for the Potentials of the Form $r^{-\lambda}$, Proceedings of the International Conference on Computational Science, 3514 (2005) 107-114.

[2] K. Srinivasan, V. Sarin, A treecode for potentials of the form $r^{-\lambda}$, International Journal of Computer Mathematics, 84 (2007) 1249-1260.

[3] K. Srinivasan, V. Sarin, A Treecode for Accurate Force Calculations, Proceedings of the International Conference on Computational Science, I (2006) 92-99.

[4] J. Barnes, P. Hut, A hierarchical O(n log n) force calculation algorithm, Nature, 324 (1986) 446-449.

[5] S.L.W. Mcmillan, S.J. Aarseth, An O(N log N) integration scheme for collisional stellar systems, The Astrophysical Journal, 414 (1993), 200-212.

[6] S. Pfalzner, P. Gibbon, Many-Body Tree Methods in Physics, The Cambridge University Press, Cambridge, Massachusetts, 1996.

[7] L. Greengard, The Rapid Evaluation of Potential Fields in Particle Systems, The MIT Press, Cambridge, Massachusetts, 1988.

[8] J. Board, K. Schulten, The fast multipole algorithm, IEEE, Computing in Science and Engineering, 2 (2000) 3-13.

[9] P. Ewald, Die Berechnung Optischer und Elektrostatischer Gitterpotentiale, Ann. Phys. 64 (1921) 253-287.

[10] R.W. Hockney, J.W. Eastwood, Computer Simulation Using Particles, McGraw-Hill, New York, 1981.

[11] C. Sagui, T. Darden, Molecular dynamics simulation of biomolecules: long-range electrostatic effects, Annu. Rev. Biophys. Biomol. Struct. 28, (1999) 155-179.

[12] E. Pollock, J. Glosli, Comments on PPPM, FMM and Ewald method for large periodic coulombic systems, Comp. Phys. Comm. 95 (1996) 93-110.

[13] T. Bishop, R. Skeel, K. Schulten, Difficulties with multiple time stepping and fast multipole algorithm in molecular dynamics, J. Comp. Chem. 18 (1997), 1785-1791.

[14] M. Deserno, C. Holm, How to mesh up Ewald sums: a theoretical and numerical comparison of various particle mesh routines, J. Comp. Phys. 109 (1998) 7678-7692.

[15] E.J. Weniger, Addition theorems as three-dimensional Taylor expansions, International Journal of Quantum Chemistry, 76 (2000) 280-295.

[16] John Avery, Hyperspherical Harmonics, Kluwer Academic Publishers, Dordrecht, Netherlands. 1989.

[17] Claus Müller, Analysis of Spherical Symmetries in Euclidean Spaces, Springer-Verlag, New York. 1991.

[18] L. Hernquist, Performance characteristics of tree codes, Astrophysics Journal Supplement, 64 (1987) 715-734.

[19] J.K. Salmon, M.S. Warren, Skeletons from the treecode closet, J. Comp. Phys. 111 (1994) 136-155.

[20] G.H. Kho, W.H. Fink, Rapidly converging lattice sums for nanoelectronic interactions, J. Comp. Chem. 23 (2001) 447-483.

[21] Y. Komeiji, M. Uebayasi, R. Takata, A. Shimizu, K. Itsukahi, M. Taiji, Fast and accurate molecular dynamics simulation of a protein using a special purpose computer, J. Comp. Chem. 18 (1997) 1546-1563.

[22] X.B. Nei, S.Y. Chen, W.N. E, M.O. Robbins, A continuum and molecular dynamics hybrid method for micro- and nano- fluid flow, J. Fluid. Mech. 500 (2004) 55-64.

[23] Z. Duan, R. Krasny, An adaptive treecode for computing nonbonded potential energy in classical molecular systems, J. Comp. Chem. 22 (2001) 184-195.

[24] I. Chowdhury, V. Jandhyala, Single level multipole expansions and operators for potentials of the form $r^{-\lambda}$, SIAM J. Sci. Comp. 26 (2004) 930-943.

[25] J. Makino, Treecode with a special-purpose processor, Astronomical Society of Japan, 43 (1991) 621-638.

[26] E.W. Hobson, The Theory of Spherical and Ellipsoidal Harmonics, New York, Chelsea Pub. Co. 1955.

[27] J. Avery, Hyperspherical Harmonics, Kulwer Academic Publishers, London, 1989.

[28] C. Müller, Analysis of Spherical Symmetries in Euclidean Spaces, Springer-Verlag, New York, 1997.

[29] E.J. Weniger, Addition theorems as three-dimensional taylor expansions, International Journal of Quantum Chemistry, 76 (2000) 280-295.

[30] V. Rokhlin, The fast multipole method: a pedestrian prescription, IEEE Trans. Antennas Propagat. 35 (1993) 7-12.

[31] L. Ying, A kernel independent fast multipole algorithm for radial basis functions. J. Comp. Phys. 213 (2006) 451-457.

# APPENDIX A

# PROOF FOR THE FIRST ADDITION THEOREM

The proofs for the following lemmas can be found in [26].

Let $f(r) = \frac{1}{r}$ and for $\theta, \phi \in \Re$, the spherical harmonics of degree $n$, $Y_n^m(\theta,\phi)$ be defined as follows:

$$Y_n^m(\theta,\phi) = \sqrt{\frac{n-|m|}{n+|m|}} P_n^{|m|}(\cos\theta) e^{im\phi}.$$

**Lemma 10.**

$$\frac{Y_n^0(\theta,\phi)}{r^{n+1}} = A_n^0 \cdot \frac{\partial^n f(r)}{\partial z^n}.$$

*For* $m > 0$*, we have*

$$\frac{Y_n^m(\theta,\phi)}{r^{n+1}} = A_n^m \cdot \left(\frac{\partial}{\partial x} + i\frac{\partial}{\partial y}\right)^m \left(\frac{\partial}{\partial z}\right)^{n-m} f(r),$$

*and*

$$\frac{Y_n^{-m}(\theta,\phi)}{r^{n+1}} = A_n^m \cdot \left(\frac{\partial}{\partial x} - i\frac{\partial}{\partial y}\right)^m \left(\frac{\partial}{\partial z}\right)^{n-m} f(r),$$

*where*

$$A_n^m = \frac{(-1)^n}{\sqrt{(n-m)! \cdot (n+m)!}}.$$

The operators $\partial_+$,$\partial_-$,$\partial_z$ are defined as follows:

$$\begin{aligned} \partial_+ &= \frac{\partial}{\partial x} + i\frac{\partial}{\partial y} \\ \partial_- &= \frac{\partial}{\partial x} - i\frac{\partial}{\partial y} \end{aligned}$$

and

$$\partial_z = \frac{\partial}{\partial z}$$

**Lemma 11.** *If $\zeta$ is a harmonic function, then*

$$\partial_+\partial_-(\zeta) = -\partial_z^2(\zeta)$$

**Theorem 12. (First Addition Theorem for Multipole Expansions)** *Let $Q = (\rho, \alpha, \beta)$ be the center of expansion of an arbitrary spherical harmonic of negative degree. Let the point $P = (r, \theta, \phi)$, with $r > \rho$ and $P - Q = (r', \theta', \phi')$. Then*

$$\frac{Y_{n'}^{m'}(\theta', \phi')}{r'^{n'+1}} = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \frac{J_m^{m'} \cdot A_n^m \cdot A_{n'}^{m'} \cdot \rho^n \cdot Y_n^{-m}(\alpha, \beta)}{A_{n+n'}^{m+m'}} \cdot \frac{Y_{n+n'}^{m+m'}(\theta, \phi)}{r^{n+n'+1}},$$

*where*

$$J_m^{m'} = \begin{cases} (-1)^{min(|m'|,|m|)} & m' \cdot m < 0 \\ 1 & m' \cdot m \geq 0 \end{cases}$$

*Proof.* From (2.1), (2.2) and Lemma 10 we have,

$$\begin{aligned}
\frac{1}{\|P - Q\|} &= \frac{1}{r'} = \sum_{n=0}^{\infty} \frac{\rho^n}{r^{n+1}} \cdot P_n(\cos\gamma) \\
&= \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \left[\rho^n \cdot Y_n^{-m}(\alpha, \beta)\right] \cdot \frac{Y_n^m(\theta, \phi)}{r^{n+1}} \\
&= \sum_{n=0}^{\infty} \left( \sum_{m=-n}^{0} \rho^n \cdot Y_n^{-m}(\alpha, \beta) \cdot A_n^m \cdot \partial_-^{|m|} \partial_z^{n-|m|} \left(\frac{1}{r}\right) \right) + \\
&\qquad \sum_{n=0}^{\infty} \left( \sum_{m=1}^{n} \rho^n \cdot Y_n^{-m}(\alpha, \beta) \cdot A_n^m \cdot \partial_+^m \partial_z^{n-m} \left(\frac{1}{r}\right) \right). \qquad \text{(A.1)}
\end{aligned}$$

We will now consider three cases.

**Case I:** $m' = 0$. From Lemma 10,

$$\frac{Y_{n'}^0(\theta', \phi')}{r'^{n'+1}} = A_{n'}^0 \cdot \partial_z^{n'} \left(\frac{1}{r'}\right). \qquad \text{(A.2)}$$

Combining (A.1) and (A.2), we obtain

$$\begin{aligned}
\frac{Y^0_{n'}(\theta',\phi')}{r'^{n'+1}} &= \sum_{n=0}^{\infty}\left(\sum_{m=-n}^{0} \rho^n \cdot Y_n^{-m}(\alpha,\beta)\cdot A^0_{n'}\cdot A^m_n \cdot \partial_-^{|m|}\partial_z^{n+n'-|m|}\left(\frac{1}{r}\right)\right) + \\
&\qquad \sum_{n=0}^{\infty}\left(\sum_{m=1}^{n} \rho^n \cdot Y_n^{-m}(\alpha,\beta)\cdot A^0_{n'}\cdot A^m_n \cdot \partial_+^{m}\partial_z^{n+n'-m}\left(\frac{1}{r}\right)\right) \\
&= \sum_{n=0}^{\infty}\sum_{m=-n}^{n}\left(\frac{\rho^n \cdot Y_n^{m}(\alpha,\beta)\cdot A^0_{n'}\cdot A^m_n}{A^m_{n+n'}}\right)\cdot \frac{Y^m_{n+n'}(\theta,\phi)}{r^{n+n'+1}},
\end{aligned}$$

where the final inequality was obtained by another application of Lemma 10.

**Case II:** $m' < 0$. Using Lemma 10 again,

$$\begin{aligned}
\frac{Y^{m'}_{n'}(\theta',\phi')}{r'^{n'+1}} &= A^{m'}_{n'}\cdot \partial_-^{|m'|}\partial_z^{n'-|m'|}\left(\frac{1}{r'}\right) \\
&= \sum_{n=0}^{\infty}\left(\sum_{m=-n}^{0} \rho^n \cdot Y_n^{-m}(\alpha,\beta)\cdot A^{m'}_{n'}\cdot A^m_n \cdot \partial_-^{|m'|+|m|}\partial_z^{n+n'-|m|-|m'|}\left(\frac{1}{r}\right)\right) + \\
&\qquad \sum_{n=0}^{\infty}\left(\sum_{m=1}^{n} \rho^n \cdot Y_n^{-m}(\alpha,\beta)\cdot A^{m'}_{n'}\cdot A^m_n \cdot \partial_-^{|m'|}\partial_+^{m}\partial_z^{n+n'-m-|m'|}\left(\frac{1}{r}\right)\right) \\
&= \sum_{n=0}^{\infty}\sum_{m=-n}^{n}\left(\frac{J^{m'}_m\cdot\rho^n \cdot Y_n^{-m}(\alpha,\beta)\cdot A^{m'}_{n'}\cdot A^m_n}{A^{m+m'}_{n+n'}}\right)\cdot \frac{Y^{m+m'}_{n+n'}(\theta,\phi)}{r^{n+n'+1}},
\end{aligned}$$

where

$$J^{m'}_m = \begin{cases} (-1)^{min(|m'|,m)} & m > 0 \\ 1 & m \le 0 \end{cases}$$

In the last inequality, for the terms with $m > 0$, Lemma 11 was used to annihilate the operators $\partial_-$ and $\partial_+$.

**Case III:** $m' > 0$. From Lemma 10,

$$\begin{aligned}
\frac{Y_{n'}^{m'}(\theta', \phi')}{r'^{n'+1}} &= A_{n'}^{m'} \cdot \partial_+^{m'} \partial_z^{n'-m'} \left( \frac{1}{r'} \right) \\
&= \sum_{n=0}^{\infty} \left( \sum_{m=-n}^{0} \rho^n \cdot Y_n^{-m}(\alpha, \beta) \cdot A_{n'}^{m'} \cdot A_n^m \cdot \partial_+^{m'} \partial_-^{|m|} \partial_z^{n+n'-|m|-m'} \left( \frac{1}{r} \right) \right) + \\
&\qquad \sum_{n=0}^{\infty} \left( \sum_{m=1}^{n} \rho^n \cdot Y_n^{-m}(\alpha, \beta) \cdot A_{n'}^{m'} \cdot A_n^m \cdot \partial_+^{m+m'} \partial_z^{n+n'-m-m'} \left( \frac{1}{r} \right) \right) \\
&= \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \left( \frac{J_m^{m'} \cdot \rho^n \cdot Y_n^{-m}(\alpha, \beta) \cdot A_{n'}^{m'} \cdot A_n^m}{A_{n+n'}^{m+m'}} \right) \cdot \frac{Y_{n+n'}^{m+m'}(\theta, \phi)}{r^{n+n'+1}},
\end{aligned}$$

where

$$J_m^{m'} = \begin{cases} (-1)^{min(m', |m|)} & m \geq 0 \\ 1 & m < 0 \end{cases}$$

Just as in Case II, we have used Lemma A.2, to annihilate the less frequent operators among $\partial_-$ and $\partial_+$. $\square$

# VITA

Kasthuri Srinivasan Kannan attended his undergraduate college at D.G.Vaishnav College, affiliated with the University of Madras, Chennai, India during 1995-98 and received his Bachelor of Science (B.Sc.) in Mathematics. He received his Master of Science in Mathematics (M.Sc.) from the Indian Institute of Technology (I.I.T.) Madras, India in 2000. Beginning in the fall of that year, he attended the Mathematics Department at Texas A&M University, College Station, as a graduate student and received his Master of Science (M.S.) in Mathematics (Computational Mathematics option) in 2002. He then joined the Computer Science Department at Texas A&M University where he received his Ph.D. in August 2008.

Currently, he is working as a research specialist at Stowers Institute for Medical Research located in Kansas City, Missouri and will continue to work with them. He can be reached at kasthuri@gmail.com. His permanent address at his home country (India) is:
No. 3/2, Indian Bank Colony
II Street, Ambattur. Chennai - 600 053
Tamil Nadu, India.

The typist for this thesis was Kasthuri Srinivasan Kannan.