Implementing Fast Multipole Methods with High-Level Interpreted Languages

Srinath Kailasa

A thesis submitted in partial fulfillment of the requirements
for the degree Master of Science

Department of Physics
University College London
August 18, 2020

Declaration

I, Srinath Kailasa, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

The Fast Multipole Method (FMM) is a numerical method to accelerate the solution of the N-Body problem, which appears in numerous contexts in science and engineering, for example in solving for gravitational or electrostatic potentials in multi-particle systems. It does so by approximating the Green's function of the system with analytic infinite series expansions (the origin of the 'multipole' in its name), and coalescing the effect of distinct distant sources together so as to greatly reduce the number of computations. The analytic expansions of the original Fast Multipole Method depend on the Green's function of the system in question (Helmholtz, Laplace etc.), and in practice a new implementation must be written for a given system.

The Kernel-Independent Fast Multipole Method (KIFMM), first presented by Ying et al. [8], is a similar approach that replaces the analytic series expansions with a continuous distribution of so called 'equivalent density' supported at discrete points on a box enclosing a set of particles. These equivalent densities are found by matching the potential they generate to those generated by the original sources at another surface in the far field. Usefully, this approach doesn't require multipole expansions of the Green's functions of a system, and therefore can be programmed in an agnostic way, hence the origin of it's name.

This thesis presents a well tested and extensible Python implementation of the KIFMM, with investigations made into both mathematical and computational techniques for the acceleration of the software, using the N-Body electrostatic problem as model on which to test the implementation. Python is chosen as it has emerged as a standard for scientific and data intensive computing in recent years, with a huge increase in adoption, and a well supported ecosystem of libraries and tools available for accelerating numerical codes. The wider context of this thesis is an ongoing collaboration with the ExaFMM Project [9] to produce a Python implementation of the KIFMM that sacrifices as little performance as possible. This thesis introduces the relevant theoretical background, before proceeding to discuss the strategies used in the practical implementation of the software. Key bottlenecks in the implementation are examined and addressed, with the speed and accuracy of the software benchmarked with the Laplace kernel for electrostatic problems. Finally, future avenues of investigation and software development are discussed, in the context of the wider literature.

Contents

Introduction

1.1 Overview of the Analytic FMM

1.1.1 Motivation

The paradigmatic problem of Fast Multipole Methods (**FMM**) [1] is the so called N-Body problem. This classic problem refers to the calculation of the pairwise interactions between N particles over a potentially long-range, for example in gravitational or electrostatic systems. The straightforward calculation can be written in the form of the following sum,

$$\Phi(x_j) = \sum_{i=1}^{N} w_i K(x_i, x_j) \tag{1.1}$$

Where $i, j \in [1, N]$ and $K(x, y)$ is called the Green's function, or equivalently a 'kernel function', where one is generally concerned with coordinates of particles in an $n = 2$ or 3 dimensional Hilbert space taking $x_i \in \mathbb{R}^n$. Additionally, each summand is weighted by w_i. For solving for electrostatic potential in three dimensions, which is used as the model problem throughout this thesis, this goes to,

$$\Phi(x_j) = \sum_{i=1}^{N} q_i K(x_i, x_j) \tag{1.2}$$

where q_i refers to a charge density with the kernel function,

$$K(x, y) = \frac{1}{4\pi\epsilon_0} \frac{1}{|x - y|} \tag{1.3}$$

the constant ϵ_0 is the permittivity of free space. It's easy to see how a naive direct application of this equation over N particles results in an algorithm of $O(N^2)$ complexity, therefore it's only practicable for systems of moderate size, whereas realistic systems, one may be interested in interactions involving 10^6 to 10^8 particles.

This chapter introduces the analytic FMM, the kernel-independent version, which is the main focus of this thesis, is presented later. Though substantially different in implementation, the analytic FMM will provide the opportunity to exposit many of the key ideas behind all FMM-based algorithms, and provides a good starting point for understanding and developing upon these algorithms. First presented by Greengard [4], the analytic FMM represented a sea change for N-Body simulation. By trading off computations for error, it manages to achieve an asymptotic complexity of just $O(N)$. Additionally, it comes equipped with rigorous error bounds, making

[1] The first usage of a technical term or abbreviation listed in the glossary is highlighted throughout the text for ease of reference.

fast and accurate massive N-Body simulations feasible on available computing hardware. It's success has been such that it is regarded as one of the key developments in numerical algorithms in the twentieth century [2].

The original analytic FMM solves the electrostatic problem in two and three dimensions, this is equivalently known as the Poisson problem, represented by the differential equation,

$$\nabla^2 \phi = f \tag{1.4}$$

Where ϕ is some scalar potential to be determined, and f is a scalar source term which is usually known. For electrostatics the corresponding formulation can be derived from Gauss' law as [6],

$$\nabla^2 \phi = -\frac{q}{\epsilon_0} \tag{1.5}$$

where ϕ is the electrostatic potential, q is the charge density and ϵ_0 is the permittivity of free space. It can therefore be seen that the FMM is actually solving the Poisson problem by reformulating it as an integral equation. The ubiquity of problems of the form (1.1) in computational science has lead to diverse application of the FMM. For example, in the modeling the electrostatic interactions of charged particles in complex biological molecules at biologically relevant length scales [1]. The extension of FMMs to Helmholtz equations [10], has lead to even more applications, such as in seismic and acoustic scattering [7]. Though as the focus of this thesis is on solving the Laplace model problem for electrostatics, this is mentioned only for completeness.

The key insight that leads to the FMM's asymptotic complexity is the idea that if the field created by a distribution of charge (or mass) density is approximated to be relatively smooth in the **far field**, then it should be possible to apply some form of compression for the evaluation of contribution to local potentials due to particles in the far field. The FMM performs this compression by encoding the field contributions of particles in the far field using a multipole expansion.

For simple kernel functions and charge distributions, such as the model problem of this thesis, one can easily derive the expression for this multipole expansion by finding an series expansion of the system's Green's function. In order to generalise the discussion, an arbitrary continuous distribution of charge is considered as shown in figure 1.1, for which the potential is evaluated at some other evaluation point outside of the distribution. This can be written as follows [6],

$$\Phi(\mathbf{r}) = \frac{1}{4\pi\epsilon_0} \int \frac{1}{d} \rho(\mathbf{r}') d\tau' \tag{1.6}$$

where $\rho(\mathbf{r}')$ is a charge density, and the other symbols take their meanings from figure (1.1).

From law of the cosines,

$$d^2 = r^2 + (r')^2 - 2rr' \cos\alpha = r^2 \left[1 + \left(\frac{r'}{r} \right)^2 - 2 \left(\frac{r'}{r} \right) \cos\alpha \right] \tag{1.7}$$

$$d = r\sqrt{1+\epsilon} \tag{1.8}$$

Where,

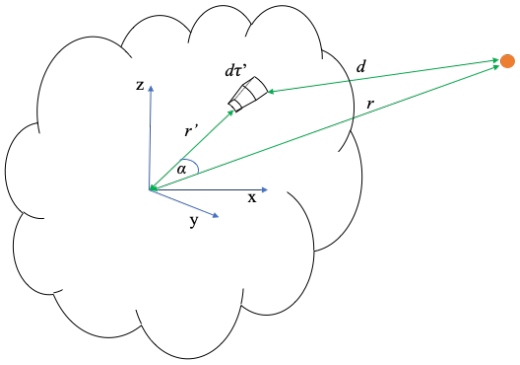

Figure 1.1: An arbitrary charge distribution, with an orange point to mark a point where the potential is being evaluated. Here, $\mathbf{r}$ is the the vector between the centre of the multipole expansion and the evaluation point, $\mathbf{r}'$ is the vector between the centre of expansion and a given volume element $d\tau'$, and d is a vector between the volume element $d\tau'$ and the evaluation point.

$$\epsilon \equiv \left(\frac{r'}{r}\right)\left(\frac{r'}{r} - 2\cos\alpha\right) \tag{1.9}$$

As ϵ is small far away from charge distribution one can expand $1/d$ binomially,

$$\frac{1}{d} = \frac{1}{r}(1+\epsilon)^{-1/2} = \frac{1}{r}\left(1 - \frac{1}{2}\epsilon + \frac{3}{8}\epsilon^2 - ...\right) \tag{1.10}$$

$$\frac{1}{d} = \frac{1}{r}\sum_{n=0}^{\infty}\left(\frac{r'}{r}\right)^n P_n(\cos\alpha) \tag{1.11}$$

Using this, the exact multipole expansion for this charge distribution is,

$$\Phi(\mathbf{r}) = \frac{1}{4\pi\epsilon_0}\sum_{n=0}^{\infty}\frac{1}{r^{n+1}}\int (r')^n P_n(\cos\alpha)\rho(\mathbf{r}')d\tau' \tag{1.12}$$

If instead we consider a charge distribution composed of N discrete charges q_i this goes to,

$$\Phi(\mathbf{r}) = \frac{1}{4\pi\epsilon_0}\sum_{i=1}^{N}\sum_{n=0}^{\infty}\frac{(r')^n q_i}{r^{n+1}}P_n(\cos\alpha) \tag{1.13}$$

Using the addition theorem for Legendre polynomials [4],

$$P_n(\cos\gamma) = \sum_{m=-n}^{n} Y_n^{-m}(\alpha,\beta)Y_n^m(\theta,\phi) \tag{1.14}$$

Where we have written the Legendre polynomial in terms of spherical harmonics, where (r,θ,ϕ) and (ρ,α,β) define two spherical coordinates, and γ is the angle subtended between them. Therefore, the multipole expansion goes to,

$$\Phi(\mathbf{r}) = \frac{1}{4\pi\epsilon_0}\sum_{i=1}^{N}\sum_{n=0}^{\infty}\frac{(r')^n q_i}{r^{n+1}}P_n(\cos\alpha) \tag{1.15}$$

$$= \frac{1}{4\pi\epsilon_0}\sum_{i=1}^{N}\sum_{n=0}^{\infty}\sum_{m=-n}^{n}\frac{(r')^n q_i Y_n^{-m}(\alpha_i,\beta_i)}{r^{n+1}}Y_n^m(\theta,\phi) \tag{1.16}$$

$$= \sum_{i=1}^{N}\sum_{n=0}^{\infty}\sum_{m=-n}^{n}\frac{M_n^m}{r^{n+1}}\cdot Y_n^m(\theta,\phi) \tag{1.17}$$

This is an exact expansion, and it converges for $\frac{r'}{r} < 1$. This convergence condition means estimating of the potential at a given evaluation point calculated using the multipole expansion is only possible in the far-field, the boundary of which is often tuned empirically for different systems as it's user defined. If instead the expansion is taken with centre at the evaluation point, one can rewrite as the multipole expansion as a 'local' expansion,

$$\Phi(\mathbf{r}) = \frac{1}{4\pi\epsilon_0}\sum_{i=1}^{N}\sum_{n=0}^{\infty}\sum_{m=-n}^{n}\frac{(r)^n q_i Y_n^{-m}(\alpha_i,\beta_i)}{r'^{n+1}}Y_n^m(\theta,\phi) \tag{1.18}$$

$$= \frac{1}{4\pi\epsilon_0}\sum_{i=1}^{N}\sum_{n=0}^{\infty}\sum_{m=-j}^{n}L_m^n\cdot Y_m^n(\theta,\phi)\cdot r^j \tag{1.19}$$

which converges when $\frac{r}{r'} < 1$. The region of convergence for both types of expansions are shown in figure (1.2).

The key point to note is that the multipole and local expansions are exact, and can be truncated as required to ensure that the asymptotic complexity of evaluating a multipole or local expansion at an evaluation point is bounded by $O(N)$. A rigorous error analysis of the analytic FMM is outside the scope of this thesis, however we refer to the literature for exact expressions for these truncation [4], Furthermore, exact operations exist for shifting the center of these expansions, which are crucial in providing the improvements to asymptotic complexity[2].

1.1.2 Algorithm Structure & Analysis

The convergence condition of the multipole expansion prohibits the compression of charges from particles in the **near field**. Therefore the FMM makes use of a tree structure in a recursive algorithm, known as an Octree in three dimensions and a Quadtree in two dimensions. This structure hierarchically partitions space such that each level, l, of the tree is equally partioned into $(2^n)^l$ boxes over the domain of the tree, where n is the dimension, i.e. $n = 3$ in three dimensions. If one were to simply traverse the tree from the coarsest level to the finest, or 'leaf', level and find the multipole expansion of source particles in each box of each level, one could then evaluate these multipole expansions at each particle to solve the N-Body problem. As there are $O(\log(N))$ boxes in the tree, and N particles this results in a $O(N\log(N))$ asymptotic complexity.

[2]Expressions for these shift operators for the three dimensional Laplace kernel considered as our model problem are provided in Appendix A.2.

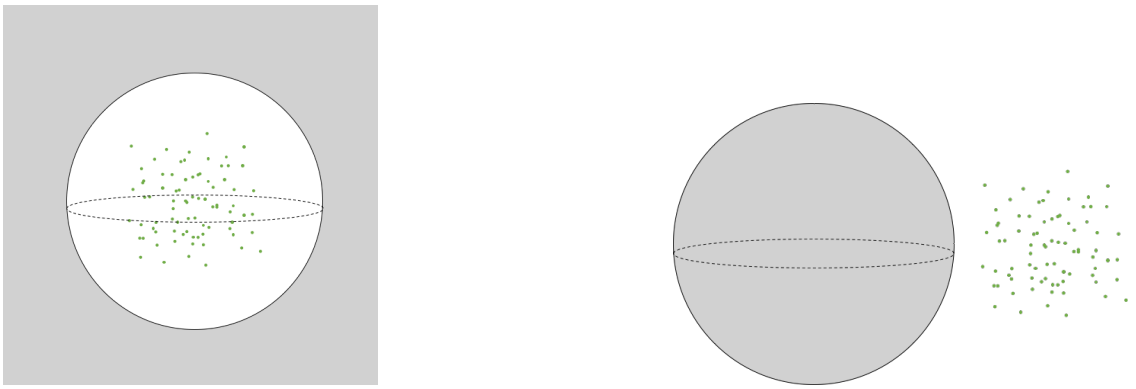

Figure 1.2: (A) A multipole expansion centered on charge distribution. (B) A local expansion, centered around a point of evaluation. Regions in which the expansions converge are shaded in grey. For the multipole expansion this is the entire domain outside of the region for $r > r'$, and for the local expansion this is the region for which $r < r'$

However the FMM reduces computational complexity further by making use of local expansions. Using the exact expressions to shift the multipole expansion coefficients[3] M_n^m to local expansion coefficients L_n^m we can approximate the the interaction between two boxes in the tree. Roughly speaking, because of the hierarchical nature of the tree, each box only needs to consider the interaction with a constant number of neighbouring boxes. Because the number of boxes $O(N)$, the FMM is bound by an $O(N)$ asymptotic complexity [7].

Using the above analysis, we can then describe the FMM algorithm in terms of two basic steps,

1. ***Upward Pass***: The tree is traversed **post-order**. Beginning at the leaf boxes, we compute a multipole expansion for each box due to the **source particles** it contains. This is also referred to as the particle-to-multipole operation, or **P2M**. Then as we move up the tree hierarchy, we compute the multipole expansions of a each box's parent box by shifting the expansion centers of the multipole expansion of a given child box, to the center of the parent box, in a multipole-to-multipole operation or **M2M**, and summing together all the contributions. Following the upward pass, one obtains the multipole expansion for each box containing source particles at all levels of the hierarchical tree.

2. ***Downward Pass***: The tree is now traversed in **pre-order**, and the local expansion of each box is computed. This local expansion is the sum of two parts: (1) the local expansion of the parent box of a given box, if it exists, which is a compression the effect of boxes non-adjacent to the current box's parent. This is also known as the local-to-local operation or **L2L**. (2) The multipole expansion of boxes which are the children of the **near neighbours** of the current box's parent but are not adjacent to the current box itself. Such boxes are described as being in the **interaction list** of a given box. This is also known as the multipole-to-local operation, or **M2L**. Notice that the M2L operation is only available from $l = 2$ of the tree, as in coarser levels the interaction list for all boxes is empty. At the end of the downward pass, when

[3]Expressions for these shift operators for the three dimensional Laplace kernel considered as our model problem are provided in Appendix A.2.

the algorithm reaches the leaf level, the local expansion of each leaf box is evaluated at all **target particles**. This local-to-particle, or **L2P**, operation encodes the far field interaction of the target particles in this leaf. This is then combined with a near field interaction, due to the source particles in the leaf box, as well as in the near neighbours, and is computed directly.

With this specification, a more detailed analysis of the algorithm is possible, though we defer to [4] for a rigorous discussion. Beginning with upward pass, at the leaf level each particle contributes to one multipole expansion. If we truncate this expansion to contain p terms, the P2M operation has a complexity of $O(Np^2)$, which can be seen from 1.17, as well as the fact that the nature of a hierarchical tree means that are $O(N)$ boxes at the leaf level. The shift operators[4] M2L, L2L and M2M require p^4 operations with this truncation, so the computation of all of these are bounded by[5] $O(Np^4)$. Finally, evaluating the p^{th} degree local expansions at each target particle in the L2P operation, is bounded by $O(Np^2)$, where the level of refinement is chosen such that there is a small constant number of κ particles at each leaf box, this leads to $O(\kappa N)$ complexity for direct calculations at the leaf level. We therefore see that the whole algorithm is bounded by $O(N)$. In practice, the number of expansion terms p is chosen for a prescribed relative error ϵ, using [6] $p = \log_c \epsilon$. The algorithm is illustrated in figure (1.3) in the two dimensional case, which is direct analogue of the three dimensional case which is the focus of this thesis, and a full pseudo-code specification is provided in Appendix A.1.

1.1.3 Summary

The main practical challenge in implementing software to solve the analytic FMM is the requirement of kernel-specific code to calculate the expansion coefficients of (1.17) and (1.19). For example, multiple different implementations already exist for Poison problems alone [5, 3]. In software engineering terms this is inconvenient as it leads to the requirement to implement problem-specific codes for each kernel one may encounter. This leads to a productivity overhead in either designing a single, extensible library, which by definition will be complex to design. Or to otherwise maintain multiple problem-specific libraries.

The algorithm thus far described referred to as the analytic FMM is more correctly called the non-adaptive analytic FMM. The non-adaptivity refers to the assumption that all boxes at the leaf level are refined to the same degree, however as mentioned in the above analysis of the algorithm, the degree of refinement is chosen such that the number of particles in a leaf box is *constant*. Therefore, If the distribution of the particles of interest is not uniform over the computational domain, one may be needlessly refining the boxes in some regions which may even be empty. Therefore, although the asymptotic complexity of the FMM is $O(N)$, it is clear that practical implementations will suffer unless care is taken to use efficient vectorised data structures for the creation of appropriate hierarchical tree required by the algorithm.

Additionally, there is significant scope for multiple levels of parallelism in practical implementations. Specifically, the computations for the local and multipole

[4]Expressions for these shift operators for the three dimensional Laplace kernel considered as our model problem are provided in Appendix A.2

[5]A more precise bound depends on the size of a given box's interaction list.

[6]There are optimal choices for c, with the authors of [8] specifying $c = \frac{4-\sqrt{3}}{\sqrt{3}}$ for three dimensional problems.

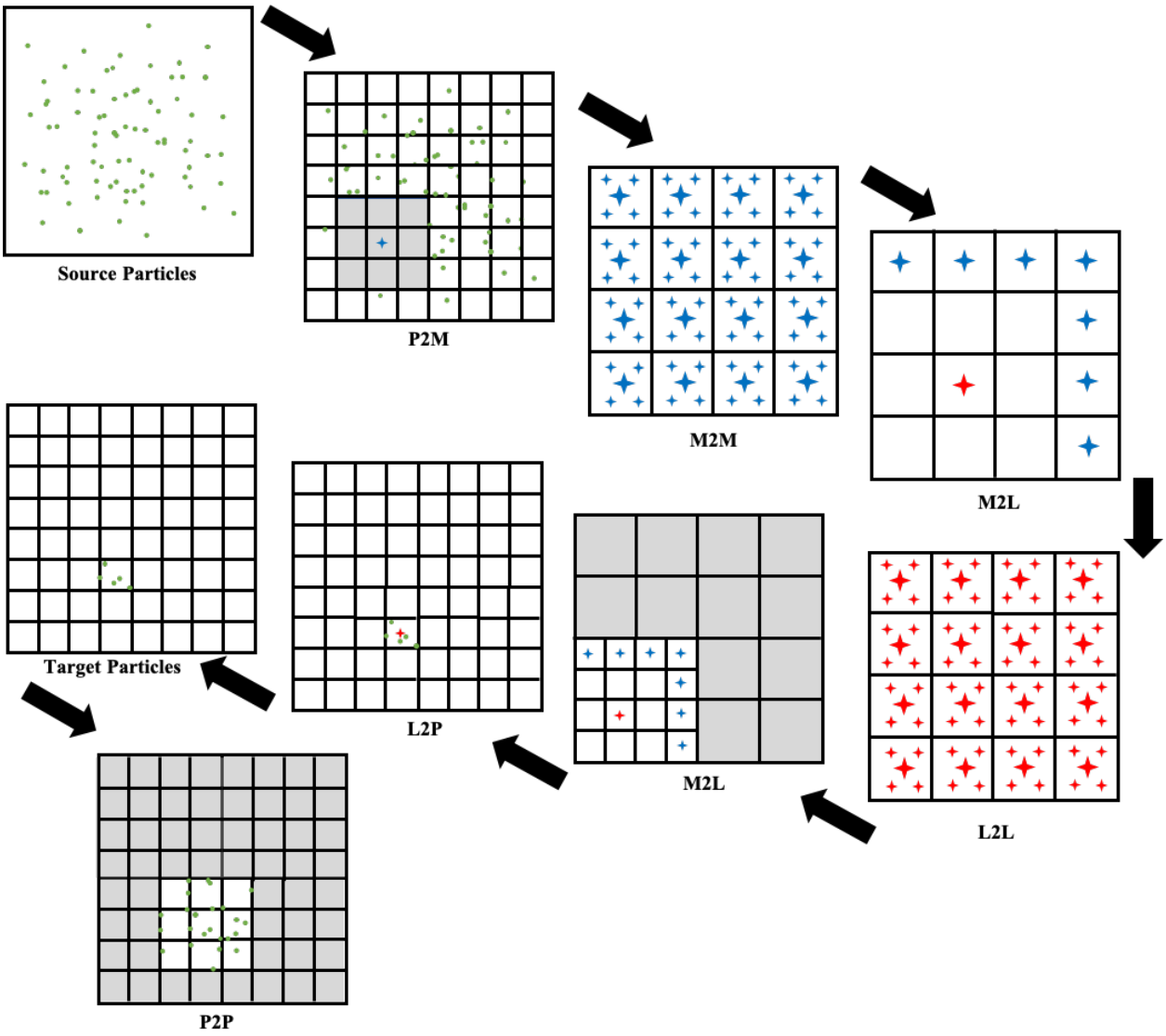

Figure 1.3: The main FMM loop in two dimensions, encapsulating the upward and downward pass. The same set of particles are used for the sources and targets, and are shown in green. Local expansions are illustrated with red stars, and multipole expansions are illustrated with blue stars. For the P2M step the grey region indicates the near field, where this multipole expansion does not converge. For the M2L and P2P steps the grey region indicates box interactions already compressed and available via the translation or direct usage of local expansions. The larger/smaller stars in the M2M and L2L steps correspond to the parent/child expansions.

coefficients as well as the application of M2L, L2L and M2M operators at a given level, are candidates for an implementation of **task-level parallelism**. In addition to parallelising of each operator application as a task, there is scope for implementing **data-level parallelism** to find the expansion coefficients. For example, Yokota and Barba [7] demonstrate how the calculation of the P2P, and M2L operators can be transferred to **GPU**s using **CUDA**.

In summary, we see that implementing the analytical FMM is complicated by the fact that it is problem specific - which will also apply to any parallel optimisation code. This in itself provides the main motivation for developing an implementation that does not rely on explicit kernel expansions. Furthermore, the desire for developer productivity, at the expense of hyper-optimised implementations, is realised in

this thesis by making use of Python, a **high level interpreted language**, for our software implementation.

1.2 Overview of the Kernel-Independent FMM

1.2.1 Motivation

Analytic FMM implementation is Kernel specific, as we need to compute the multipole expansion coefficients. It's instead based solely on kernel evaluations, therefore can be programmed in an agnostic way.

1.2.2 Algorithm Structure & Analysis

1.2.3 Summary

Strategy for Practical Implementation

2.1 Bottleneck Analysis

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

2.2 Space-Filling Curves

2.3 Operator Caching

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

2.4 SVD Compression

2.5 Software Design

Experiments & Results

3.1 Section 1

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam

vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. å

Conclusion

4.1 Section 1

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Appendix

A.1 FMM Algorithm Specification

This pseudo-code is adapted from [4].

Initialisation: Choose a level of refinement of $n \approx \log_8 N$ and precision ϵ, set $p = \lceil -\log_2(\epsilon) \rceil$[1].

Upward Pass

Step 1

Form multipole expansions of potential field due to particles in each box at leaf level.

do $ibox = 1, .., 8^l$
Form p^{th} degree multipole expansion for each leaf box.
end

Step 2

Translate Multipole expansion to coarser levels from the bottom up.

do $l = n - 1, .., 0$
do $ibox = 1, .., 8^n$
Perform M2M operations.
end
end

Downward Pass

Computations at the coarsest possible level. For a given box, done by including interactions with those boxes which are well separated, and whose interactions have not been accounted for at the parent level.

Step 3

Form local expansion about center of each box at each level $l \leq n - 1$, describes field due to all particles that are not contained in the current box, it's near neighbours or it's secondary near neighbors.

[1] Different authors have different suggestions for p, the authors of [8] use $p = \log_c \epsilon$ with $c = \frac{4-\sqrt{3}}{\sqrt{3}}$

do $l = 1, ..., n-1$
  do $ibox = 1, ..., 8^l$
    Perform M2L translations.
  end
  do $1, ..., 8^l$
    Perform L2L operations.
  end
end

Step 4

After this step, local expansions are available at the leaf level. One can use this to evaluate potential at leaves from all particles in the far field.

do $ibox = 1, ..., 8^n$
  Find local expansion at leaf level, by doing M2L from interaction list.
end

Step 5

Evaluate local expansions at particle positions in all leaves

do $ibox = 1, ..., 8^n$
  For every particle in *ibox*'th box, evaluate local expansion.
end

Step 6

Compute nearest neighbors directly,
do $ibox = 1, ..., 8^n$
  For every particle in *ibox*'th box, compute potential directly with nearest neighbors.
end

Step 7

do $ibox = 1, ..., 8^n$
  Add direct and far field terms together for every particle in the *ibox*
end

A.2 Analytic FMM Operators for 3D Laplace Kernel

The expressions presented here are first derived in [4]

For l charges of strengths $q_1, ..., q_l$ located inside sphere D of radius a center at $Q = (\rho, \alpha, \beta)$, and that for points $P = (r, \theta, \phi)$ outside D potential given by the following **Multipole Expansion**,

$$\Phi(P) = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \frac{O_n^m}{r'^{n+1}} \cdot Y_n^m(\theta', \phi')) \tag{A.1}$$

Where the points P and Q are defined such that $P - Q = (r', \theta', \phi')$. Then for any point P outside sphere D_1 of radius $a + \rho$, The multipole expansion can shifted with the following **M2M operation**,

$$\Phi(P) = \sum_{j=0}^{\infty} \sum_{k=-j}^{j} \frac{M_j^k}{r^{j+1}} \cdot Y_j^k(\theta, \phi) \tag{A.2}$$

Where,

$$M_j^k = \sum_{n=0}^{j} \sum_{m=-n}^{n} \frac{O_{j-n}^{k-m} \cdot J_m^{k-m} \cdot A_n^m \cdot A_{j-n}^{k-m} \cdot \rho^n \cdot Y_n^{-m}(\alpha, \beta)}{A_j^k} \tag{A.3}$$

and,

$$J_m^{m'} = \begin{cases} (-1)^{\min(|m'|,|m|)} & \text{if } m \cdot m' < 0 \\ 1 & \text{otherwise} \end{cases} \tag{A.4}$$

$$A_n^m = \frac{(-1)^n}{\sqrt{(n-m)! \cdot (n+1)!}} \tag{A.5}$$

For l charges of strengths $q_1, ..., q_l$ located inside sphere D_Q of radius a center at $Q = (\rho, \alpha, \beta)$, and that $\rho > (c+1)a$ with $c > 1$. The corresponding multipole expansion, converges inside sphere D_0 of radius a centered at origin. Inside D_0 the potential due to charges has the local expansion,

$$\Phi(P) = \sum_{j=0}^{\infty} \sum_{k=-j}^{j} L_j^k \cdot Y_j^k(\theta, \phi) \cdot r^j \tag{A.6}$$

where,

$$L_j^k = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \frac{O_n^m \cdot J_k^m \cdot A_n^m \cdot A_j^k \cdot Y_{j+n}^{m-k}(\alpha, \beta)}{A_{j+n}^{m-k} \cdot \rho^{j+n+1}} \tag{A.7}$$

where A_n^m same as above,
but,

$$J_m^{m'} = \begin{cases} (-1)^{n'}(-1)^{\min(|m'|,|m|)} & \text{if } m \cdot m' < 0 \\ (-1)^{n'} & \text{otherwise} \end{cases} \tag{A.8}$$

n' refers to j, m' refers to k

and that for points $P = (r, \theta, \phi)$ outside D potential given by multipole expansion. This is also known as the multipole to local, or **M2L Operation**.

Let $Q = (\rho, \alpha, \beta)$ be the origin of a local expansion,

$$\Phi(P) = \sum_{n=0}^{p} \sum_{m=-n}^{n} O_n^m \cdot Y_n^m(\theta', \phi') \cdot r'^n \tag{A.9}$$

Where $P = (r, \theta, \phi)$ and $P - Q = (r', \theta', \phi')$.

$$\Phi(P) = \sum_{j=0}^{p} \sum_{k=-j}^{j} L_j^k \cdot Y_j^k(\theta, \phi) \cdot r^j \tag{A.10}$$

where,

$$L_j^k = \sum_{n=j}^{p} \sum_{m=-n}^{n} \frac{O_n^m \cdot J_{n-j,m-k}^m \cdot A_{n-j}^{m-k} \cdot A_j^k \cdot Y_{n-j}^{m-k}(\alpha, \beta) \cdot \rho^{n-j}}{A_j^k} \tag{A.11}$$

where A_n^m same as above,

$$J_{n,m}^{m'} = \begin{cases} (-1)^n(-1)^m & \text{if } m \cdot m' < 0 \\ (-1)^n(-1)^{m'-m} & \text{if } m \cdot m' > 0 \text{ and } |m'| < |m| \\ (-1)^n & \text{otherwise} \end{cases} \tag{A.12}$$

This operation defines the local to local, or **L2L Operation**.

Glossary

CUDA 'Compute Unified Device Architecture', is a parallel computing API designed for the development of programs for **GPU**s . 9

data-level parallelism In a multiprocessor system, data level parallelism is achieved by distributing data amongst different compute nodes to be executed upon in parallel. See **CUDA** . 9

far field An set of particles in the far field are considered to be far away enough from the particle of interest that they are suitably described by a convergent multipole expansion. . 4, 8

FMM The Fast Multipole Method. . 1, 3, 4

GPU 'Graphics Processing Unit', are specialised processors specifically designed for rapid parallel execution across large blocks of data. . 9

high level interpreted language A High-Level language is one that offers an API and primitive data structures that strongly abstract from the details of the computer on which it is being run. An interpreted language is one in which codes are run directly via an *interpreter*, rather than first being compiled. In reality interpreted languages run in a 'virtual machine', which takes input code, transforms it to byte-code which is then translated into machine level instructions. This approach allows for greater portability of code, and developer productivity, at the expense of space and time overhead in running software. . 10

interaction list Boxes which are the children of the near-neighbours of the a box's parent box, but are not adjacent to the box itself. . 7, 8

KIFMM The 'Kernel Independent' Fast Multipole Method. . 1

L2L Translate from local of parent box to local expansion of child box.. 7, 8

L2P Evaluate local of leaf box at each target particle in a leaf box.. 8

M2L Translate from multipole of box A to local expansion of box B.. 7, 8

M2M Translate from multipole of child box to multipole expansion of parent box.. 7, 8

near field An set of particles in the near-field are considered to violate the the criteria for describing them with a convergence multipole expansion. . 6, 8, 9

near neighbours Two boxes in computational tree are near neighbours if they are at the same level of refinement, and share a boundary point. . 7, 8, 16

P2M Particle to multipole expasion operation.. 7, 8

post-order In reference to the traversal of heirarchical trees, post-order traversal is moving from the finest (leaf) level to the coarsest (root) level. . 7

pre-order In reference to the traversal of heirarchical trees, pre-order traversal is moving from the coarsest (root) level to the finest (leaf) level. . 7

source particles Particles are described as sources if they are consifered to contribute to the source field being evaluated at the target particles. They may or may not refer to the same set of particles as the target particles, for example in the classic N-Body problem, the target particles and the source particles are the same set. . 7

target particles Particles are described as targets if the potential due to a source field is being evaluated at these particle's positions, but they themselves are not being considered as a source for the field being evaluated. They may or may not refer to the same set of particles as the source particles, for example in the classic N-Body problem, the target particles and the source particles are the same set. . 8

task-level parallelism Task-level parallelism is achieved when multiple threads or processes, running the same, or differing, code, in a multiprocessing system are executed with the same, or differing, data. . 9

well separated Two boxes in computational tree are near neighbours if they are at the same level of refinement, and are not near neighbours. . 16

Bibliography

[1] John A. Board et al. "Accelerated molecular dynamics simulation with the parallel fast multipole algorithm". In: *Chemical Physics Letters* 198.1 (1992), pp. 89 –94. ISSN: 0009-2614. DOI: https://doi.org/10.1016/0009-2614(92)90053-P. URL: http://www.sciencedirect.com/science/article/pii/000926149290053P.

[2] Barry A. Cipra. "The Best of the 20th Century: Editors Name Top 10 Algorithms". In: *SIAM News* 33.4 (2000).

[3] Frank Ethridge and Leslie Greengard. "A New Fast-Multipole Accelerated Poisson Solver in Two Dimensions". In: *SIAM J. Sci. Comput.* 23.3 (Mar. 2001), 741–760. ISSN: 1064-8275. DOI: 10.1137/S1064827500369967. URL: https://doi.org/10.1137/S1064827500369967.

[4] Leslie Greengard. "The Rapid Evaluation of Potential Fields in Particle Systems". PhD thesis. Yale University, 1987.

[5] Leslie Greengard and June Yub Lee. "A direct adaptive poisson solver of arbitrary order accuracy". English (US). In: *Journal of Computational Physics* 125.2 (May 1996), pp. 415–424. ISSN: 0021-9991. DOI: 10.1006/jcph.1996.0103.

[6] David J. Griffiths. *Introduction to Electrodynamics*. 4th ed. Cambridge University Press, 2017. DOI: 10.1017/9781108333511.

[7] Wen-Mei W. Hwu. *GPU Computing Gems Emerald Edition*. 1st. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011. ISBN: 0123849888.

[8] Denis Zorin Lexing Ying George Biros. "A kernel-independent adaptive fast multipole algorithm in two and three dimensions". In: *Journal of Computational Physics* 196.2 (2004), pp. 591–626. DOI: http://dx.doi.org/10.1016/j.jcp.2003.11.021.

[9] Lorena A. Barba Rio Yokota. *ExaFMM User's Manual*. 2011. URL: http://www.bu.edu/exafmm/files/2011/06/ExaFMM-UserManual1.pdf.

[10] Vladimir Rokhlin. "Rapid Solution of Integral Equations of Theory in Two Dimensions". In: *Journal of Computational Physics* 86.2 (Feb. 1990), pp. 414–439. DOI: 10.1016/0021-9991(90)90107-C.