Hierarchical Particle Mesh: An FFT-accelerated Fast Multipole Method

Nickolay Y. Gnedin1,2,3

1 Fermi National Accelerator Laboratory; Batavia, IL 60510, USA; gnedin@fnal.gov
2 Kavli Institute for Cosmological Physics; The University of Chicago; Chicago, IL 60637 USA
3 Department of Astronomy & Astrophysics; The University of Chicago; Chicago, IL 60637 USA

Received 2019 April 21; revised 2019 May 30; accepted 2019 June 17; published 2019 July 23

Abstract

I describe a modification to the original Fast Multipole Method (FMM) of Greengard & Rokhlin that approximates the gravitation field of an FMM cell as a small uniform grid (a “gridlet”) of effective masses. The effective masses on a gridlet are set from the requirement that the multipole moments of the FMM cells are reproduced exactly, hence preserving the accuracy of the gravitational field representation. The calculation of the gravitational field from a multipole expansion can then be computed for all multipole orders simultaneously, with a single Fast Fourier Transform, significantly reducing the computational cost at a given value of the required accuracy. The described approach belongs to the class of “kernel independent” variants of the FMM method and works with any Green function.

Key words: methods: numerical

1. Introduction

The Fast Multipole Method (FMM) invented by Greengard & Rokhlin (1987, 1997) is deservedly quoted as one of the top 10 computational algorithms of the 20th century (Cipra 2000). It is the only known method for computing self-gravity of a collection of bodies (“particles”) that has asymptotically linear scaling with the number of interacting particles. It does, however, have a bottleneck; the key step of the algorithm—the computation of the gravitational field from the multipole expansion—is rather expensive. Not surprisingly, significant effort has been put into accelerating this step of the algorithm.

The FMM algorithm relies on the subdivision of the whole computational domain into separate cells organized in an oct-tree in three dimensions or a quad-tree in two dimensions. The FMM algorithm and its variant described here are not dimensionally specific; however, for the sake of presentation, whenever a specific value for the dimension of space is needed, I will choose the three dimensions case as the most common case for astrophysical applications. While there is significant freedom in specifying the details of the underlying tree, in this paper I only consider a cubic (square in two dimensions) computational domain and cubic oct-tree cells. With this restriction, the whole computational domain forms the single cell on level 0 (or root level). The level 0 cell is divided in three dimensions into eight “child” cells (into four cells in two dimensions and two cells in one dimension) and becomes their “parent” cell. Each of the child cells can, in turn, be divided into eight more child cells, etc. The resultant tree may or may not be of uniform depth, hence supporting Adaptive Mesh Refinement.

The algorithm itself consists of four main steps. In the first step, the gravitational field (expressed either as gravitational potential or accelerations due to gravity) of each childless cell—a “leaf”—is computed and parameterized with a finite number of parameters. In the original FMM formulation of Greengard & Rokhlin (1987, 1997), these parameters are the multipoles of the matter distribution, but other forms of parameterizations have been used as well, and it is also the key ingredient of the Hierarchical Particle Mesh (HPM) method described in this paper. In the second step, the parameterizations of the child cells are combined into a parameterization of their parent cell for all cells in the tree. In the original FMM formulation, this step was called “multipole-to-multipole,” or M2M, translation. To be more general and to allow for non-multipole parameterizations, I am going to call it the “child-to-parent transformation.” At the end of the child-to-parent transformation, each cell in the tree has a parameterized approximation to the gravitational field created by all the matter inside it.

The third step is the key step of the algorithm and relies on the subdivision of space around a given cell in the tree (I will call it a “target” cell) into two zones: the “near zone” consisting of all cells that immediately neighbor the target cell even by a single point (there are, at most, 26 of such cells in three dimensions and fewer if some of the neighbor cells are not refined as deep as the target cell). The rest of the domain is assigned to the “far zone.” All the cells in the far zone that belong to the near zone of the target cell parent but do not belong to the near zone of the target cell itself are labeled as an “interaction zone.”

Figure 1 shows a sketch of such subdivision of space in two dimensions (the three-dimensional case is completely analogous). The contribution to the gravitational field everywhere inside of the target cell is composed of three contributions.

1. The field from the near zone and the target cell itself is computed explicitly, either by direct summation or by some other approach, such as nonperiodic Fast Fourier Transform (FFT).
2. The contribution from the interaction zone is computed explicitly using the gravitational field parameterization in the source cells of the interaction zone. In the original FMM terminology this step is called “multipole-to-local” or M2L transformation and reduces to simply computing the gravitational field from the multipole expansion from all interaction zone cells. I will call this step “source-to-target transformation” to allow for non-multipole parameterizations.
3. The contribution from the rest of the far zone is added through the target cell parent, grandparent, grand-grandparent, etc., whose interaction zone contributions...
summed together make the whole of the rest of the far zone. In fact, if these contributions are accumulated as the tree is traversed from level 0 up (I use the convention when the more refined levels are located “higher” in the tree, as the tree grows “up”), when the parent of a given target cell is ready to add its contribution to the gravitational potential of the target cell, the parent gravitational field already includes all the contributions from the grandparent, grand-grandparent, etc., all the way down to the root of the tree. I call this last, fourth step of the algorithm the “parent-to-child” transformation in contrast to the original FMM terminology of “local-to-local” or L2L translation.

The FMM algorithm is then applied to all leaf cells in the computational domain (or all cells if the gravitational field is required to be known on non-leaf cells too).

The important limitation of the original FMM algorithm is now apparent. In three dimensions, there are up to $6^3 = 216$ source cells in the interaction zone, and computing multipole contributions from all of them one by one is going to form a computational bottleneck. It may be more efficient if all these contributions are computed simultaneously, and that is the main feature of the algorithm described below.

2. HPM Method

Multipole representation of the gravitational field in a target cell from a source cell in the interaction zone is simply a convenient parameterization. A different parameterization would work equally well as long as it provides an equally accurate approximation to the gravitational field. For example, a set of potential values at discrete points on a sphere or cube surrounding the target and source cell to parameterize the internal (for a target cell) or external (for a source cell) gravitational fields have been used extensively in the past (see Makino 1999; Kawai & Makino 2001; Ying et al. 2004; Ying 2006; Rogers 2015; March & Biros 2017). Other parameterizations, such as Chebyshev polynomials (Fong & Darve 2009) or weights of Gaussian quadratures (Létourneau et al. 2010), have also been used.

In the HPM method, the gravitational field is parameterized by effective masses arranged on a small regular grid inside each cell, as shown in Figure 2: I call such a small grid a “gridlet” in order to distinguish it from the global grid or from a grid covering the interaction zone introduced below. In this terminology, each cell of the FMM tree has an associated unique gridlet and each gridlet covers one, and only one, cell.

Gridlet-like representation of the multipole expansion and its associated advantage of being able to use FFT is not new and has been used in the past in FMM and non-FMM multipole-based approaches (Berman 1995; Ong et al. 2003; Ying 2006; Liska & Colonius 2014; Nitadori 2014). The key feature of the HPM method is the use of a single FFT for computing the source-to-target transformation for all effective masses (or, equivalently, for all multipole orders) and all cells in the interaction zone at once, and the associated gain in efficiency with FFT padding reduced from the standard factor of 2 to only a factor of $4/3$, which is described below.

Of the previously discussed algorithms, the most similar to HPM is the method described by Liska & Colonius (2014). HPM would reduce to Liska & Colonius (2014) approach in the special case of (a) uniform distribution of particles on a lattice, and (b) the number of effective masses equal to the number of particles, with each effective mass equal to the corresponding particle mass, and (c) using cell-by-cell FFT for cells in the interaction zone instead of a single full interaction zone FFT. I comment on the relative performance of the two approaches below. A possibility of using FFT for accelerating the source-to-target transformation was also mentioned by Berman (1995) but was not described or discussed.

2.1. Source-to-target (“Multipole-to-local”) Transformation

Let the number of effective masses on a gridlet along one dimension be $N_g$ ($N_g^D$ effective masses in total for a D-dimensional case). Given the distribution of effective masses on the gridlets in the interaction zone, one can compute the source-to-target transformation (“multipole-to-local” transformation in terminology of Greengard & Rokhlin 1987, 1997) for all interaction zone cells at once if one constructs a $(6N_g^D)$ grid of effective masses in the whole interaction zone (dark blue points in Figure 3—the size of the interaction zone is always $6^3$ FMM tree cells, as can be seen from Figure 1) and solves for the value of the potential in the target cell gridlet (orange points...
in Figure 3) with a standard particle mesh (PM) method (see Hockney & Eastwood 1981) using an FFT, although with two key caveats. First, the choice of the Green function for the PM solver matters. Using a Green function for a finite difference representation of the Poisson equation would be inaccurate, since the PM solver, in that case, would retain the low spatial resolution of the gridlet. Instead, one should use the exact Green function for the 1/r point mass potential so that the potential values on the gridlet are exact and the multipoles computed from these values are accurate up to the same order as the source multipoles.

Second, and most importantly, the interaction zone does not have periodic boundary conditions, so one needs to use a nonperiodic PM solver. The simplest approach to nonperiodic FFT is to double the grid size and to pad the added volume with zeros. For example, for the two dimensional case with \( N_g = 2 \) shown in Figure 3, one can use a grid of the size of \( N_{tot} = 2 \times 6 \times N_g = 24 \) along one dimension. The exact Green function for the Poisson solver can then be constructed as the Fourier transform of the \( N_{tot} \times N_{tot} \) array \( G_{ij} = log(|l_j|^2 + |l_i|^2)/(4\pi) \), where \( |l_i| \) denotes the periodic distance between index \( i \) and the origin:

\[
|l_i| = \begin{cases} i, & \text{for } 0 \leq i \leq N_{tot}/2 \\ N_{tot} - i, & \text{for } N_{tot}/2 < i < N_{tot}. \end{cases}
\]

The value at \( i = 0 \) and \( j = 0 \) is the “softened” value of the potential; in the tests below, I set it to the value at \( i = 1 \) and \( j = 0 \). Obviously, in three dimensions

\[
G_{ijk} = \frac{1}{4\pi(|l_{ij}| + |l_{jk}| + |l_{ki}|)^2}.
\]

An immediate and important optimization can be obtained if one notices that the largest one dimension (i.e., along-the-axis) distance between the edge of the target cell and any edge of any cell in the interaction zone is only four, so the Fourier transform only needs to be done in a grid of the size of \( N_{tot} = 8 \times N_g \), not \( N_{tot} = 12 \times N_g \). In that case, some of the potential values for gridlets in the interaction zone would be incorrect, but those are discarded anyway (the implied inefficiency of computing and discarding some values is discussed in the next paragraph). Even higher optimization is possible if nonperiodic FFT is done on the original \( (6 \times N_g)^D \) grid without any padding. One such method was described by James (1977; see also Magorrian 2007). Unfortunately, the James’ method is not suitable for HPM, since it is essentially based on the finite difference representation of the Poisson equation rather than on using the exact Green function (1). A new method for nonperiodic FFT with the exact Green function will indeed provide a significant optimization for HPM.

Another potential optimization can be considered if one notices that the potential only needs to be computed inside the target cell, i.e., in a small fraction of the FFT grid. However, this gives only a very modest saving because the forward FFT from the real space to Fourier space needs to be done in the full \( (8N_g)^3 \) grid (three passes over the data), the multiplication by the Green function also needs to be done in the full grid (one pass over data), and in the backward transform, the first one dimension transform over \( k_x \) of the Fourier image, \( \Phi(k_x, k_y, k_z) \), also needs to be done in full (one pass over data). Only after that the transform over \( k_y \) of the intermediate data, \( \Phi(k_x, y, z) \), can be done for \( 1/8 \) of the full \( z \)-range, and the last transform of \( \Phi(k_x, y, z) \) can be done over 1/64 of the full data pass. Hence, the total operation count for such an optimization is \( 3 + 1 + 1 + 1/8 + 1/64 \approx 5.14 \) versus the total operation count without such optimization of \( 7 \), resulting in saving of about 27%, which is probably too little to be worthy of the additional complex bookkeeping.

### 2.2. Defining Effective Masses

The key to the accuracy of the HPM method is the correct assignment of the effective masses on gridlets. For example, using a standard nearest-grid-point or cloud-in-cell assignment of particle masses into the gridlet density would result in a highly inaccurate approximation to the combined gravitational field (in a language of multipoles, only the monopole and the dipole are preserved in such an assignment).

Instead, for a given gridlet size \( N_g \), one can require that the combined gravitational field of \( N_g^D \) effective masses is identical to the gravitational field of \( N_g^D \) multipoles. This is directly analogous to the definition of “pseudo-particles” by Makino (1999). It is more convenient to use Cartesian multipoles in this case, since spherical multipoles in the original Greengard & Rokhlin (1987, 1997) formulation are generally complex numbers, and by using Cartesian multipoles one also avoids complex and computationally expensive operations with spherical harmonics.

Below, I use concise and clear notation of Visscher & Apalkov (2010), although Cartesian multipoles were introduced earlier (see Shimada et al. 1994; Challacombe et al. 1996). Given a density distribution inside an FMM cell, \( \rho(r) \), with boldface notation used to denote spatial vectors, one can define Cartesian multipoles as

\[
Q_n = \frac{1}{m} \int \rho(r) r^n dr.
\]
generalization (or, more exactly, reduction) of the equations to one-dimensional and two-dimensional cases is always trivial.

Effective masses, \( M_p \), in a gridlet with \( p = (i, j, k) \) and \( i, j, \) and \( k \) spanning the range from 0 to \( N_g - 1 \) can now be defined so that the Cartesian multipoles inside the gridlet up to and including the order of \( N_g - 1 \) are faithfully reproduced, i.e.,

\[
Q_n = \frac{1}{n!} \sum_{p=0}^{N_g-1} M_p r_p^n.
\]

For a gridlet of the physical size \( L \),

\[
r_p^n = L^n x_p^n,
\]

where \( L^n \equiv L^{n_1+n_2+n_3} \) and \( x_p \) are constants

\[
x_p = \left( \frac{i + 1/2}{N_g} - \frac{1}{2}, \frac{j + 1/2}{N_g} - \frac{1}{2}, \frac{k + 1/2}{N_g} - \frac{1}{2} \right),
\]

and hence

\[
\sum_{p=0}^{N_g-1} M_p x_p^n = \frac{n!}{L^n} Q_n.
\]

If we treat the vector-valued indices \( p \) and \( n \) as “raveled” sequential indices spanning the range from 0 to \( N_g^3 - 1 \), then the left-hand side of Equation (4) is just a matrix multiplication with the constant matrix \( A_{np} \equiv x_p^n \). Matrix \( A_{np} \) is a Vandermonde matrix, has a nonzero determinant, and has a well-defined inverse \( B_{pn} \equiv A_{np}^{-1} \), which can be precomputed for the whole simulation. Hence, the first step of the FMM algorithm of computing the effective masses on gridlets reduces to computing the Cartesian multipoles and a matrix multiplication:

\[
M_p = \sum_n B_{pn} \frac{n!}{L^n} Q_n,
\]

which is only insignificantly more computationally expensive than the original FMM method.

2.3. Child-to-parent (“Multipole-to-multipole”) Translation

The second step of the FMM algorithm of computing effective masses on the parent cell gridlet from effective masses on child cell gridlets is also straightforward. From Equation (9) of Visscher & Apalock (2010),

\[
Q_p^n = \sum_m \frac{e^m}{m!} Q_{C}^{n-m},
\]

where \( Q_p^n \) are Cartesian multipoles in the parent cell and \( Q_{C}^{n-m} \) are Cartesian multipoles in a child cell whose center if offset from the center of the parent cell by vector \( c \). From Equations (3) and (5), it follows that

\[
M_p^n = \sum_n B_{pn} \frac{n!}{L^n} \sum_m \frac{e^m}{m!} \frac{L^{n-m}}{(n-m)!} \sum_q A_{(n-m)q} M_q
\]

\[
= \sum_q T_{pq} M_q,
\]

and matrix

\[
T_{pq} = \sum_n \sum_m B_{pn} \frac{n!}{m! (n-m)!} \frac{(a^C)^{n-m}}{2^n} A_{mq},
\]

describes the transformation from a child effective masses to the parent effective masses for a child identified by vector \( a^C \), which is the location of the center of the child cell relative to the parent cell in units where the parent cell goes from \(-1\) to \(1\), \( a^{C-1} = (-1/2, -1/2, -1/2) \) for the first child, \( a^{C-2} = (+1/2, -1/2, -1/2) \) for the second child, etc. Matrices \( T_{pq} \) are constant and can be precomputed for the whole simulation.

2.4. Parent-to-child (“Local-to-local”) Translation

The final step of the FMM algorithm, the translation of potential values from the parent gridlet to the child gridlet, now becomes almost trivial. The target-to-source transformation using FFT transforms effective masses, \( M_p \), from the interaction zone cells into the potential values, \( \Phi_p \), on the gridlet of the target cell. Given \( \Phi_p \), one can create a polynomial approximation for the potential anywhere inside the cell associated with the gridlet by computing the potential multipoles first,

\[
V_n = \sum_p B_{pn} \Phi_p,
\]

and then using them for the polynomial approximation for the potential,

\[
\Phi(x) = \sum_n V_n x^n.
\]

Given a polynomial approximation for the parent cell, \( \Phi^P(x) \), parent contribution to the potential values on the child gridlet are reduced to a simple evaluation:

\[
\Phi^C = \Phi^P \left( \frac{a^C + x_p}{2} \right).
\]

This step is, therefore, identical to the original FMM method with an additional matrix multiplication in Equation (6).

2.5. Possible Modifications

One can also consider several straightforward modifications of the HPM method. For example, since the FFT provides a solution inside the entire cube that includes the interaction zone, including the whole of the near zone, one can consider using the potential values not only inside the target cell gridlet (orange points in Figure 3), but also outside the target cell (gray points in Figure 3). If one defines such an extended gridlet that is widened on all sides by just one extra point, the main optimization discussed in Section 2.1—using \((8 N_g^3)\) grid for the Fourier transform rather than naïvely expected \((12 N_g^3)\)—remains valid, because in the FFT algorithm, the Nyquist frequency is sampled correctly, i.e., for an \( n \)-long FFT, all wavenumbers from 0 to \( n/2 + 1 \) are not affected by periodic boundary conditions; for the optimized HPM with \( n = 8 N_g^3 \), there are \( 4 N_g^3 + 1 \) such wavenumbers, and the gravitational potential on all gray points in Figure 3 is computed correctly.

One may expect that having more potential values in the target gridlet, and, in particular, including values outside of the target cell, would improve the accuracy of interpolation in Equation (7). This is indeed so; however, the improvement is only modest, since the error in the potential from the interaction zone is dominated by the finite number of effective masses (equal to the number of multipoles) in the source cells. Tests
show that the modest gain in accuracy is compromised by the higher computational cost of a wider gridlet.

Another possible modification that I explored is using an alternative method for potential interpolation in the target cells. Given the potential values, \( \Phi_p \), in the target gridlet, one can, for example, use spline interpolation instead of the polynomial interpolation of Equation (7). I tried using B-splines for such interpolation, but again found no improvement.

In fact, the polynomial interpolation achieves higher accuracy than splines unless the spline order is comparable to the polynomial order. While this may sound counterintuitive, one should remember that interpolation from Equation (7) is actually the Taylor expansion of the exact gravitational potential around the center of the target cell, and the error term in Equation (7) scales as the polynomial of degree \((N_n + 1)\). Spline interpolation must have the same order for the error term to be competitive.

One may wonder whether a different FFT-based approach may be more efficient—namely using FFT to compute source-to-target transformation separately for each cell in the interaction zone with appropriately precomputed Green functions. Liska & Colonius (2014) take this approach. With packing, such an approach is not competitive. Indeed, for the full \((8N_n)^3\) grid, the operation count is \((8N_n)^3 \log_2((8N_n)^3) = 1536N_n^3 \log_2(N_n)\), ignoring the prefactor in the FFT operation count of \(N \log_2 N\), which is irrelevant for the relative comparison. Each cell-by-cell packed FFT would take \((2N_n)^3 \log_2((2N_n)^3)\) operations, and with up to 189 of such cell-by-cell FFTs, the total operation count becomes \(4536N_n^3 \log_2(2N_n)\). The ratio of the full grid FFT to 189 cell-to-cell FFTs is then

\[
0.34 \frac{p + 3}{p + 1},
\]

where \(p = \log_2 N_n\). For a monopole approximation \((p = 0)\), the two approaches are comparable, but for any higher order scheme, the full grid FFT wins over cell-by-cell ones.

However, if one uses a nonperiodic FFT with the exact Green function without padding, cell-by-cell FFTs become more competitive. In that case, the full grid FFT count is \((6N_n)^3 \log_2(6N_n)^3) = 648N_n^3(p + 1.6)\), while cell-by-cell FFTs add up to \(189N_n^3 \log_2(N_n^3) = 567N_n^3(p + 1)\), which always beats the full grid FFT, albeit by only about 13%.

Irrespective of whether the single full grid FFT or cell-by-cell FFTs are used, a nonperiodic FFT with the exact Green function and without padding would lead to a substantial speedup of the HPM method by at least a factor of 1536/648 \(\approx 2.4\).

3. Comparison with the Original FMM

3.1. Operation and Memory Scaling

The first and the last steps of the HPM algorithm differ from their original FMM counterparts by only adding a matrix multiplication. Let \(M = N_n^2\) be the total number of multipoles and \(K\) the average number of particles (for a particle code) or grid cells (for a grid code) in each FMM tree cell. In the original FMM computing the multipoles from particles/grid cells (the first step of FMM) and computing the gravitational field from multipoles onto particles/grid cells both require \(\approx K \times M\) multiply-add operations if all spherical harmonics can be precomputed and cached—this is possible for a grid code, but is not possible for a particle code, which has a higher operation count depending on how spherical harmonics are computed.

In the HPM method, an extra contribution of \(\approx M^2\) multiply-add operations is added due to matrix multiplications. Since the most likely use case for the FMM algorithm is \(K \gg M\), the extra matrix multiplication in its HPM variant does not add any significant overhead. Even for \(K \sim M\), the overhead is insignificant since these two steps contribute only a small fraction of the overall computational cost. Both algorithms have the same memory requirements (\(M\) multipoles per FMM tree cell), since matrix \(B_p\) in Equations (5) and (6) is constant and can be precomputed once and for all.

The second step of the HPM algorithm (child-to-parent transformation) is identical to its original FMM counterpart and requires \(M^2\) operations and no extra memory.

The key difference between the original FMM and HPM is the third: the source-to-target transformation. The original FMM algorithm of Greengard & Rokhlin (1997) would require at least \(5 \times 189 \times M^2 \approx 950M^2\) multiply-add operations (in a grid code with precomputed spherical harmonics, and even more for a particle code)—the factor of five in front comes from the operation count in Equation (5.6) of Greengard & Rokhlin (1997). In order to speed up this most expensive step of the original FMM algorithm, Greengard & Rokhlin (1997) and Cheng et al. (1999) advocated rotating coordinate frames in which multipoles are computed by aligning the multipole axes with the \(z\)-axis before transforming multipoles from the source cell to the target cell (Equation (23) of Cheng et al. 1999). Such rotation changes the operation count of the source-to-target transformation to about \(2800M^{3/2}\) operations and becomes an optimization for multipoles past the octupole.

The HPM algorithm as presented requires only about \((8N_n)^3 \log_2((8N_n)^3) = 512M \log_2(512M)\) operations if the FFT implementation is efficient. If the FFT is done “in place,” both the original FMM and HPM require no extra memory.

3.2. Performance

For comparison with the original FMM method (Greengard & Rokhlin 1987, 1997), I use a simple implementation of the HPM algorithm with the source-to-target transformation computed on a \((8 \times N_n)^3\) grid (I present a comparison for the three-dimensional case, as it is the most common one) using the FFTW solver version 3.3.8 (http://fftw.org). In order to achieve high precision, the FFT needs to be done in double precision, and all other gridlet values (Cartesian multipoles, \(Q_n\) and \(V_n\); effective masses, \(M_g\); and the gridlet potentials, \(\Phi_p\)) are also kept in double precision. It appears that FFTW works faster in double precision on a 64-bit machine, so going to single precision would not be an optimization.

Since the Poisson equation is linear, it is sufficient to only consider the case of a point source. As the code framework for the comparison, I use the “PEx” prototype (Gnedin et al. 2018). PEx implements the oct-tree of small cubic grids (“patches”) and solves for gravity using the FMM algorithm on the tree of patches, with each patch being a single “cell” in the FMM tree. Because PEx is the prototype for the hydrodynamic code, each patch in these tests is a grid of \(K = 8^3\) or \(K = 16^3\) hydrodynamic cells—the particular value for the patch size is unimportant, the relative performance of the HPM and the original FMM does not depend on the patch size.
The timing of the source-to-parent transformation is not expected to depend significantly on the grid geometry or the level of refinement of the FMM tree. To test for that, I perform three different tests: with global grid (1) uniformly refined to four levels ($2^4 \times 8 = 128^3$ underlying uniform grid of hydrodynamic cells), (2) uniformly refined to five levels ($256^3$ uniformly refining uniform grid of hydrodynamic cells), and (3) uniformly refined to three levels with the first octant ($0 \leq x, y, z \leq 1/2$) refined to level four and a region ($1/8 \leq x, y, z \leq 3/8$) refined to level five. The accuracy of the numerical solution is measured against the analytically known point-source solution for the actual accelerations (the gradient of the potential) with the point source placed at the location (1,1,1); the error is computed as a relative error. The HPM algorithm uses $2^3$, $4^3$, $6^3$, and $8^3$ gridlets, while the original FMM is run with the maximum spherical harmonic number from 1 to 15 with an increment of 2.

In Figure 4, I show timing for three different algorithms: the original Greengard & Rokhlin version of FMM (Greengard & Rokhlin 1997), the original FMM with multipole rotations, and the HPM flavor of the FMM algorithm, as described above, versus two different error norms for gravitational accelerations. The HPM method outperforms the original FMM with multipole rotation by a factor of 3–5 in wall-clock time at a given accuracy or, equivalently, by about an order of magnitude in accuracy for a given wall-clock time.

4. Conclusions

The computational performance numbers should be taken with care, as they depend on the specific software implementation. The PEx implementations of all three algorithms are reasonably optimized, with all factors that can be precomputed (spherical harmonics and Wigner matrices in the original FMM at all hydrodynamic cells in a patch and exact Green function and matrices, $A_{pq}$, $B_{pq}$, and $T_{pq}$ for the HPM method) and tabulated at the start of the simulation and before the onset of timing (i.e., in all three implementations, only the operations that explicitly depend on the values of the matter density inside each FMM cell are performed in the timed section of the code, and all other factors, including those that depend only on geometry, such as $Y_{n}^{m}(\theta, \phi)$, are precomputed and tabulated). All needed temporary storage is also pre-allocated before the timed section of the code. Nevertheless, no code could not be optimized further.

Another important consideration of any algorithm is parallelization. In both the original FMM and HPM, all operations are done per FMM tree cell, except in the source-to-target transformation, when multipoles from 189 source cells are needed for each target cell. Hence, the same parallelization strategy would work for both algorithms.

The HPM algorithm has another important advantage over the original FMM: since the FFT in the source-to-target transformation is used just to compute the convolution, any Green function in place of Equation (1) would work equally well. Hence, HPM belongs to the class of “kernel independent” variants of the FMM.

I am thankful to Volker Springel and Andrey Kravtsov for valuable suggestions and references to related prior work and to the anonymous referee for the constructive comments that significantly improved the original manuscript. This manuscript has been authored by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics.

References

Berman, C. 1995, SIAM J. Sci. Comput., 16, 1082
Challaconbe, M., Schwegler, E., & Almlöf, J. 1996, JChP, 104, 4685
Cheng, H., Greengard, L., & Rokhlin, V. 1999, JCoPh, 155, 468
Cipra, B. A. 2000, SIAM News, 33, 1
Fong, W., & Darve, E. 2009, JCoPh, 228, 8712
Gnedin, N. Y., Semenov, V. A., & Kravtsov, A. V. 2018, JCoPh, 359, 93
Greengard, L., & Rokhlin, V. 1987, JCoPh, 73, 325
Greengard, L., & Rokhlin, V. 1997, AChNum, 6, 229
Hockney, R. W., & Eastwood, J. W. 1981, Computer Simulation Using Particles (Abingdon: Taylor & Francis)
James, R. A. 1977, JCoPh, 25, 71
Kawai, A., & Makino, J. 2001, ApJL, 550, 5143
Létourneau, P.-D., Cecka, C., & Darve, E. 2010, Materials Sci. Engineer. Conf. Ser., 10, 012230
Liska, S., & Colonius, T. 2014, JCoPh, 278, 76
Magorrian, J. 2007, MNRAS, 381, 1663
Makino, J. 1999, JCoPh, 151, 910
March, W. B., & Biros, G. 2017, Appl. Comput. Harmonic Analysis, 43, 39
Nitadori, K. 2014, arXiv:1409.5981
Ong, E., Lim, K., Lee, K., & Lee, H. 2003, JCoPh, 192, 244
Rogers, D. M. 2015, JChPh, 142, 074101
Shimada, J., Kaneko, H., & Takada, T. 1994, JCoCh, 15, 28
Visscher, P. B., & Apalkov, D. M. 2010, JMMM, 322, 275
Ying, L. 2006, JCoPh, 213, 451
Ying, L., Biros, G., & Zorin, D. 2004, JCoPh, 196, 591