New Parallel Optimal-Parameter Fast Multipole Method (OPFMM)

CHEOL HO CHOI,1 KLAUS RUEDENBERG,2 MARK S. GORDON2

1Department of Chemistry, College of Natural Sciences, Kyungpook National University, Taegu 702-701, South Korea
2Department of Chemistry and Ames Laboratory, Iowa State University, Ames, Iowa 50011

Received 28 December 2000; accepted 19 March 2001

Dedicated to Professor Paul von R. Schleyer

ABSTRACT: The three key translation equations of the fast multipole method (FMM) are deduced from the general polypolar expansions given earlier. Simplifications are introduced for the rotation-based FMM that lead to a very compact FMM formalism. The optimum-parameter searching procedure, a stable and efficient way of obtaining the optimum set of FMM parameters, is established with complete control over the tolerable error ($\varepsilon$). This new procedure optimizes the linear scaling with respect to the number of particles for a given $\varepsilon$. In addition, a new parallel FMM algorithm, which requires virtually no internode communication, is suggested that is suitable for the parallel construction of Fock matrices in electronic structure calculations. © 2001 John Wiley & Sons, Inc.

Keywords: optimum-parameter searching; fast multipole method

Introduction

Since Greengard and Rokhlin1 first introduced the fast multipole method (FMM), which has the potential to reduce the $O(N^2)$ work of calculating a pairwise potential to $O(N)$ for $N$ particles, many implementations of their method have been suggested. Schmidt and Lee2 showed that it is practical and efficient to carry out the FMM in 3 dimensions for finite and periodic systems. Zhao and Johnsson3 reported a fast implementation of the 3-dimensional nonadaptive parallel multipole method on a connection machine. Greengard and Gropp4 also discussed parallelizations of the algorithm. Board et al.5 implemented serial and parallel versions of FMM that facilitated large protein molecular dynamics simulations on workstations. The use of Cartesian multipoles was suggested6 for very large-scale molecular dynamics simulations in which high order expansions are not necessary. Pérez-Jordá and Yang7 introduced real spherical harmonics to improve their recursive bisection method.8

The very FMM (VFMM),9 which uses smaller ex-
pansions for distant cells in the interaction list, was shown to improve the speed by about 2–3 times for 3-dimensional systems without sacrificing accuracy.

In the meantime, many quantum chemists recognized the potential use of FMM in quantum mechanical computations in order to overcome traditional high scaling barriers. Their studies yielded many encouraging results, such as quantum chemical tree code (QCTC), \(^\text{10}\) continuous FMM, \(^\text{11}\) and Gaussian VFMM (GvFMM). \(^\text{12}\) Goedecker's \(^\text{13}\) recent review provides more complete information.

White and Head-Gordon contributed many significant improvements, \(^\text{14}\) such as the rotation-based FMM and the fractional tier method. These authors demonstrated that linear scaling with respect to the number of particles can be achieved as long as edge effects can be minimized. However, current methods do not have direct explicit control over the error in FMM calculations. Aluru et al. \(^\text{15}\) argued that the complexity of the FMM is distribution dependent and therefore does not scale linearly. Pérez-Jordá and Yang \(^\text{16}\) supported this argument. However, it is based on the premise that the multipole order must increase with the system size. In reality, because the machine precision is finite, the accuracy of FMM computations cannot be improved infinitely by simply increasing the multipole order. As discussed in detail in the following section, one way to overcome the machine precision is to increase the well-separateness parameter. Further, to date there is no analytical assessment of the relation between FMM performance and accuracy. This is a complex problem due to the intricately interrelated FMM parameters, so the problem is not simply assessing FMM accuracy but assessing FMM accuracy at its best performance level.

Current interest derives from the potential use of FMM methods in the parallel construction of the Fock matrix in electronic structure computations. To make the FMM robust for such uses, it is important to develop an approach that permits explicit control over the error in the method.

The present article addresses several new aspects in the derivation, formulation, and parallel implementation of FMM and shows that linear scaling with complete control over error can be achieved. An alternative scaling of solid spherical harmonics that leads to more efficient translation formulas for the rotation-based FMM is introduced. A new parallel algorithm requiring virtually no internode communication is presented, as is a stable and efficient way to obtain the optimal set of FMM parameters. Finally, the performance of the new parallel FMM is illustrated.

### Scaled Solid Spherical Harmonics

The FMM for the calculation of electrostatic potentials and energies is based on expansions in terms of spherical harmonics. Considerable computational efficiencies are achieved by exploiting and manipulating certain compact algorithms for the translation of spherical harmonics between spatial centers and the rotation of spherical harmonics around spatial centers. We begin with a reexamination of these algorithms that leads to a new scaling of the solid spherical harmonics.

#### CONVENTIONAL REGULAR AND IRREGULAR SOLID HARMONICS

Using the spherical coordinates
\[
\vec{r} = (x, y, z) = r (\sin \theta \cos \phi, \sin \theta \sin \phi, \cos \theta),
\]
and Condon and Shortley’s phase conventions, \(^\text{17}\) we choose the complex surface spherical harmonics in the form
\[
Y_{lm}(\theta, \phi) = (-1)^m P^m_l(\cos \theta) e^{im\phi} / \sqrt{2\pi}
\]
where the normalized Legendre functions are those given by Bethe \(^\text{18}\)
\[
P^m_l(t) = \left[ \frac{(2l+1)(l-m)!}{2(l+m)!} \right]^{1/2} P^m_l(t),
\]
with \(t = \cos \theta\) and the standard associated Legendre functions being
\[
P^m_l(t) = \frac{1}{2l+1} \frac{d}{dt} \left( 1 - t^2 \right)^{m/2} \frac{d^{l+m}}{dt^{l+m}}(t^2 - 1)^l.
\]
Defined in this way, normalized Legendre functions with positive and negative \(m\) values are related by
\[
P^{-m}_l(t) = (-1)^m P^m_l(t)
\]
Solid spherical harmonics are the fundamental solutions of Laplace’s equation of potential theory in spherical coordinates. They are of two kinds, which are traditionally defined \(^\text{19}\) as regular solid harmonics \([r^l Y^m_l(\theta, \phi)]\) and irregular solid harmonics \([r^{-l-1} Y^m_l(\theta, \phi)]\). Although the former are convergent at the origin and singular at infinity, the converse holds true for the latter.
TRANSLATIONS OF MODIFIED SOLID HARMONICS

A thorough discussion of the translations of solid harmonics between different centers was given by Steinborn and Ruedenberg. They showed that all translation formulas are most concisely expressed in terms of modified regular ($\hat{Y}^m_l$) and irregular solid spherical harmonics ($\hat{Z}^m_l$), which they defined as follows:

$$\hat{Y}^m_l(\mathbf{r}) = r^l Y_{lm}(\theta, \phi) \left[ \frac{2l+1}{4\pi} \frac{(l-m)!(l+m)!}{(2l+1)!} \right]^{1/2},$$

(6)

$$\hat{Z}^m_l(\mathbf{r}) = \frac{1}{r^{l+1}} Y_{lm}(\theta, \phi) \left[ \frac{4\pi}{2l+1} (l-m)!(l+m)! \right]^{1/2}. \quad (7)$$

Identical definitions were subsequently adopted by White and Head-Gordon who termed these functions the “scaled associated Legendre polynomials.”

Steinborn and Ruedenberg showed that all special formulas can be deduced as special cases from two general “polypolar expansions.” For the modified regular solid harmonics the expansion is

$$\hat{Y}^M_l(\mathbf{r}_1 + \mathbf{r}_2 + \cdots + \mathbf{r}_n) = \sum_{l_1m_1} \sum_{l_2m_2} \cdots \sum_{l_nm_n} \hat{Y}^m_{l_1}(\mathbf{r}_1) \times \hat{Y}^m_{l_2}(\mathbf{r}_2) \cdots \hat{Y}^m_{l_n}(\mathbf{r}_n) \quad (8)$$

where the indices run over the ranges defined by

$$0 \leq l_i \leq L, \quad -l_i \leq m_i \leq l_i, \quad i = 1, 2, 3, \ldots, n,$$

(9)

and are additionally restricted by the conditions

$$L = \sum_{i=1}^n l_i, \quad M = \sum_{i=1}^n m_i. \quad (10)$$

For the modified irregular solid harmonics the expansion is

$$\hat{Z}^M_l(\mathbf{r}_1 + \mathbf{r}_2 + \cdots + \mathbf{r}_{n+1}) = \sum_{l_1m_1} \sum_{l_2m_2} \cdots \sum_{l_nm_n} (-1)^{l_1+m_1+l_2+m_2+\cdots+l_n+m_n} \times \hat{Y}^{-m_1}_{l_1}(\mathbf{r}_1) \hat{Y}^{-m_2}_{l_2}(\mathbf{r}_2) \cdots \hat{Y}^{-m_n}_{l_n}(\mathbf{r}_n) \times \hat{Z}^{M+m_1+m_2+\cdots+m_n}_{l_1+l_2+\cdots+l_n}(\mathbf{r}_{n+1}) \quad (11)$$

where the indices run over ranges defined by

$$0 \leq l_i \leq \infty, \quad -l_i \leq m_i \leq l_i, \quad i = 1, 2, \ldots, n,$$

(12)

and the vectors $\mathbf{r}_k$ satisfy

$$|\mathbf{r}_1 + \mathbf{r}_2 + \mathbf{r}_3 + \cdots + \mathbf{r}_n| < |\mathbf{r}_{n+1}|. \quad (13)$$

The argument of the irregular solid harmonic on the right-hand side is thus the longest vector. (A further discussion of conditions regarding the arguments is given in ref. 20.) By virtue of the parity properties of the spherical harmonics,

$$\hat{Y}^m_l(-\mathbf{r}) = (-1)^l \hat{Y}^m_l(\mathbf{r}), \quad (14)$$

$$\hat{Y}^m_l^* = (-1)^m \hat{Y}^{-m}_l(\mathbf{r}), \quad (15)$$

expansion (11) can also be written as

$$\hat{Z}^M_l(\mathbf{r}_{n+1} - \mathbf{r}_1 - \mathbf{r}_2 - \cdots - \mathbf{r}_n) = \sum_{l_1m_1} \sum_{l_2m_2} \cdots \sum_{l_nm_n} \left[ \hat{Y}^m_{l_1}(\mathbf{r}_1) \hat{Y}^m_{l_2}(\mathbf{r}_2) \cdots \hat{Y}^m_{l_n}(\mathbf{r}_n) \right]^* \times \hat{Z}^{M+m_1+m_2+\cdots+m_n}_{l_1+l_2+\cdots+l_n}(\mathbf{r}_{n+1}) \quad (16)$$

where the asterisk indicates the complex conjugate.

For low values of $n$ and/or $L$, the general polypolar expansions reduce to various older formulas. Thus, in the case of the regular solid harmonics, $n = 2$ yields the “second addition theorem”

$$\hat{Y}^M_l(\mathbf{r}_1 + \mathbf{r}_2) = \sum_{l_1m_1} \sum_{l_2m_2} \sum_{l_3m_3} \sum_{l_4m_4} \hat{Y}^m_{l_1}(\mathbf{r}_1) \hat{Y}^m_{l_2}(\mathbf{r}_2) \hat{Y}^m_{l_3}(\mathbf{r}_1) \hat{Y}^m_{l_4}(\mathbf{r}_2) \quad (17)$$

where the indices $l_i$ and $m_i$ are restricted by the conditions

$$L = l_1 + l_2, \quad M = m_1 + m_2. \quad (18)$$

If $\mathbf{r}_1$ and $\mathbf{r}_2$ are chosen as illustrated in Figure 1a, eq. (17) represents the translation of a regular solid harmonic from the atomic center $B$ to the atomic center $A$ in a molecule and $P$ in Figure 1a could represent the position of an electron.

For the irregular harmonics, the case $n = 1$ of the identity (16) can be written in the form

$$\hat{Z}^M_l(\mathbf{r}_1 - \mathbf{r}_2) = \sum_{l_1m_1} \sum_{l_2m_2} \sum_{l_3m_3} \sum_{l_4m_4} \hat{Y}^m_{l_1}(\mathbf{r}_1) \hat{Y}^m_{l_2}(\mathbf{r}_2) \hat{Z}^{M+m_1+m_2+m_3+m_4}_{l_1+l_2+l_3+l_4}(\mathbf{r}_3) \quad (19)$$

where we have put $\mathbf{r}_1 = \mathbf{r}_1' \mathbf{r}_2 = \mathbf{r}_2'$, with $\mathbf{r}_1'$ denoting the shorter and $\mathbf{r}_2'$ the longer of vectors $\mathbf{r}_1$ and $\mathbf{r}_2$. As can be seen from Figure 1b, expansion (19), which proves useful in the FMM, can be looked at in two ways. The value at $C$ of an irregular harmonic centered at $B$ is expressed in terms of irregular harmonics centered at $A$, which is further from $C$ than from $B$ (the expansion coefficients being regular harmonics); or the value at $B$ of an irregular harmonic centered at $C$ is expressed in terms of regular harmonics centered at $A$, which is closer to $B$ than to $C$ (the expansion coefficients being irregular harmonics).
FIGURE 1. The spatial arrangements of the argument vectors in the solid harmonics for various special cases of the polypolar expansions (see text).

Noting on the other hand that for \( l = 0 \) eq. (7) yields \( \hat{Z}_0^L(\vec{r}) = 1/|\vec{r}| \), one obtains from eq. (16)

\[
|\vec{r}_{n+1} - \vec{r}_2 - \vec{r}_1|^{-1} = \sum_{l_1 m_1} \sum_{l_2 m_2} \sum_{l_3 m_3} \sum_{l_{n+1} m_{n+1}} \left[ \hat{Y}_{l_1}^{m_1}(\vec{r}_1) \hat{Y}_{l_2}^{m_2}(\vec{r}_2) \cdots \hat{Y}_{l_{n+1}}^{m_{n+1}}(\vec{r}_{n+1}) \right]^* \\
\times \hat{Y}_{l_{n+1}}^{m_{n+1}}(\vec{r}_{n+1}) \hat{Y}_{l_1}^{m_1}(\vec{r}_1) \hat{Y}_{l_2}^{m_2}(\vec{r}_2) \cdots \hat{Y}_{l_{n+1}}^{m_{n+1}}(\vec{r}_{n+1}).
\]

(20)

The case \( n = 2 \) can be written as

\[
|\vec{R} + \vec{r}_2 - \vec{r}_1| = \sum_{l_1 m_1} \sum_{l_2 m_2} \sum_{l_3 m_3} \sum_{l_4 m_4} \left[ \hat{Y}_{l_1}^{m_1}(\vec{r}_1) \hat{Y}_{l_2}^{m_2}(\vec{r}_2) \hat{Y}_{l_3}^{m_3}(\vec{r}_3) \hat{Y}_{l_4}^{m_4}(\vec{r}_4) \right]^* \\
\times \hat{Y}_{l_1}^{m_1}(\vec{r}_1) \hat{Y}_{l_2}^{m_2}(\vec{r}_2) \hat{Y}_{l_3}^{m_3}(\vec{r}_3) \hat{Y}_{l_4}^{m_4}(\vec{r}_4).
\]

(21)

which is Carlson and Rushbrooke’s bipolar expansion of the electrostatic potential between two unit charges at points \( P \) and \( Q \) when \( \vec{r}_3 = \vec{R} \) and \( \vec{r}_1, \vec{r}_2 \) are chosen as illustrated in Figure 1c with respect to the two centers \( A \) and \( B \).

In the case of \( n = 1 \) one obtains

\[
|\vec{r}_3 - \vec{r}_1| = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \hat{Y}_l^m(\vec{r}_1) \hat{Y}_l^m(\vec{r}_3) \\
= \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \frac{(l+m)!}{l! m!} \frac{r_1^l}{r_3^l} P_l^m(\cos \theta) \\
\times P_l^m(\cos \theta) e^{i(m \phi_1 - m \phi_3)}.
\]

(22)

As illustrated in Figure 1d, the left-hand side is the electrostatic potential at point \( P \), which is due to a unit point charge at \( Q \), or at point \( Q \), which is due to a charge at \( P \), and the right-hand side is Laplace’s monopolar expansion of this potential around center \( A \).

**ROTATION OF MODIFIED AND SCALED SOLID HARMONICS**

Translations of solid harmonics take very simple explicit forms when the \( z \) axis is parallel to the translation vector so that the latter has spherical coordinates \( \theta = \phi = 0 \) in the coordinate system of the harmonics. White and Head-Gordon therefore suggested that it is efficient to first rotate the spherical harmonics such that their \( z \) axes are parallel to the translation axis. The translation is then performed along the \( z \) axis, after which the solid harmonics are rotated back to the original coordinate system. In such a context, the efficient rotation of solid harmonics is therefore also a very relevant consideration.

Now, a rotation of the coordinate system induces a unitary transformation among the surface harmonics, namely,

\[
Y_L^m(\vec{r}) = \sum_{k=-L}^{L} D_{LM}^L R (\theta, \phi) Y_L^m(\vec{r})
\]

(23)

where the \( D_{LM}^L \) are the Wigner rotation matrices that transform the spherical harmonics defined with respect to one coordinate system \( \vec{r} \) into the spherical harmonics defined with respect to the rotated coordinate system \( \vec{r}' \). An efficient and numerically stable recurrence procedure for the rapid evaluation of the matrices \( D_L^L \) directly from the elements of the coordinate rotation matrix \( R \) was recently given by Choi et al.\textsuperscript{23}

From eq. (23) one obtains the following rotation formulas for the modified solid harmonics \( \hat{Y}_L^m(\vec{r}) \) and \( \hat{Z}_L^m(\vec{r}) \):

\[
\hat{Y}_L^m(\vec{r}) = \sum_{k=-L}^{L} \left( \frac{\sqrt{(L-k)!}(L+k)!}{\sqrt{(L-|k|)!(L+|k|)!}} \right) D_{LM}^L Y_L^m(\vec{r})
\]

\[
\hat{Z}_L^m(\vec{r}) = \sum_{k=-L}^{L} \left( \frac{\sqrt{(L-|k|)!(L+|k|)!}}{\sqrt{(L-|k|)!(L+|k|)!}} \right) D_{LM}^L \hat{Z}_L^m(\vec{r})
\]

(24a)

(24b)

These two formulas contain different clumsy factors. Because two rotations but only one special translation along the \( z \) axis are required for a general movement of the solid harmonics, it is manifestly preferable to have simple rotation formulas rather than a simple translation formula. In addition, the summation in the rotation formulas goes from \(-L\) to \( L \) whereas (as shown below) only half the number of terms are needed in the translation formulas.
We therefore found it advantageous for the numerical efficiency of the FMM procedure to define alternative modified regular ($\tilde{Y}_L^m(\bar{r})$) and irregular ($\tilde{Z}_L^m(\bar{r})$) solid harmonics:

$$\tilde{Y}_L^m(\bar{r}) = r^l Y_{lm}(\theta, \phi)^+ \left[ \frac{4\pi}{2l+1} \right]^{1/2} (-1)^m \left[ \frac{(l-m)!}{(l+m)!} \right]^{1/2} r^l P^m_l(\cos \theta)e^{-im\phi} \quad (25)$$

and

$$\tilde{Z}_L^m(\bar{r}) = \frac{1}{r^{l+1}} Y_{lm}(\theta, \phi) \left[ \frac{4\pi}{2l+1} \right]^{1/2} (-1)^m \left[ \frac{(l-m)!}{(l+m)!} \right]^{1/2} \frac{1}{r^{l+1}} P^m_l(\cos \theta)e^{im\phi}. \quad (26)$$

With these definitions the rotated $\tilde{Y}_L^M(\bar{r})$ and $\tilde{Z}_L^M(\bar{r})$ are obtained by the simple equations

$$\tilde{Y}_L^M(\bar{r}) = \sum_{k = -L}^L D_{LM}^k \tilde{Y}_L^k(\bar{r}) \quad (27a)$$

$$\tilde{Z}_L^M(\bar{r}) = \sum_{k = -L}^L D_{LM}^k \tilde{Z}_L^k(\bar{r}) \quad (27b)$$

which are identical to those for the surface harmonics without additional numerical prefactors.

We shall call these alternate modified solid spherical harmonics the scaled solid harmonics.

**TRANSLATIONS OF SCALED SOLID HARMONICS**

In the FMM procedure developed here we shall need the formulas for translating the scaled solid harmonics of eqs. (25) and (26) along the z axis. Specifically, three relations provided the operational basis for our algorithm.

Consider first the second addition theorem, eq. (17). Applying the condition $\theta_1 = \phi_1 = 0$ to eq. (17) and recasting it in terms of the new harmonics of eqs. (25) and (26), one obtains the "$y2y$" translation for the scaled regular solid harmonics:

$$\tilde{Y}_L^M(\bar{r}_1 + \bar{r}_2) = \sum_{j = |M|}^L |\bar{r}_1|^{l-j} (L + M)! (L - M)! (j + M)! (j - M)! \right)^{1/2} \times \tilde{Y}_j^M(\bar{r}_2) \quad (28)$$

This describes the shift of the expansion center from $B$ to $A$ when $R = \bar{r}_1$ in Figure 1a is chosen to be along the z axis.

Next, consider eq. (19). Setting $\theta_\ast = \phi_\ast = 0$, implying that $\bar{r}_\ast$ lies along the z axis, and inserting the scaled harmonics, one obtains the "$y2z$" translation

$$\tilde{Y}_L^M(\bar{r}_\ast - \bar{r}_\ast) = \frac{(-1)^{l+M}}{[(L - M)! (L + M)!]^{1/2}} \times \sum_{l = |M|}^\infty \frac{|\bar{r}_\ast|^{l+M} (l - M)! (l + M)!} {[(l-M)! (l+M)!]^{1/2}} \tilde{Y}_L^M(\bar{r}_\ast)^\ast \quad (29)$$

which shifts the expansion center from $B$ to $A$. The smaller the ratio $|\bar{r}_\ast|^{-1} / |\bar{r}_\ast|$, the more rapidly the sum in eq. (29) converges. Note that, in practice, the summation is limited to the highest order of $L$.

On the other hand, setting $\theta_\ast = \phi_\ast = 0$ in eq. (19), implying that $\bar{r}_\ast$ lies along the z axis, and inserting the scaled harmonics, one obtains the "$z2z$" translation

$$\tilde{Z}_L^M(\bar{r}_\ast - \bar{r}_\ast) = \frac{1}{[(L - M)! (L + M)!]^{1/2}} \times \sum_{l = |M|}^\infty \frac{|\bar{r}_\ast|^{l+M} (l - M)! (l + M)!} {[(l-M)! (l+M)!]^{1/2}} \tilde{Z}_L^M(\bar{r}_\ast) \quad (30)$$

which shifts the expansion center from $A$ to $C$.

The three translation formulas given in eqs. (28), (29), and (30) in conjunction with the rotation formulas in eqs. (27a) and (27b) are the key equations required to implement the FMM procedure to be described in the subsequent sections.

**CHARGE-WEIGHTED SCALED SOLID SPHERICAL HARMONICS**

Sometimes it is convenient to define charge-weighted scaled regular ($\tilde{Y}_L^m(q)$) and irregular ($\tilde{Z}_L^m(q)$) spherical harmonics,

$$\tilde{Y}_L^m(q)(\bar{r}) = q \tilde{Y}_L^m(\bar{r}) \quad (31)$$

$$\tilde{Z}_L^m(q)(\bar{r}) = q \tilde{Z}_L^m(\bar{r}), \quad (32)$$

where $q$ represents a point charge. Then the interaction energy between a charge $q_A$ located at $A$ and a charge $q_B$ located at $B$ is given by

$$q_A q_B |\bar{r}_\ast - \bar{r}_\ast|^{-1} = \sum_{l = 0}^\infty \sum_{m = -l}^l \tilde{Y}_L^m(\bar{r}_\ast) \tilde{Z}_L^m(\bar{r}_\ast)^\ast \quad (33)$$

The charge-weighted scaled harmonics of eqs. (31) and (32) manifestly satisfy the same translation relations as the scaled harmonics: the former can be substituted for the latter in eqs. (28), (29), and (30) without any modifications. This is relevant because the translations of charge-weighted scaled harmonics are particularly useful in the context of the FMM algorithm to be discussed.
Parallel Fast Multipole Algorithm

Pairwise Coulomb interactions among classical point charges require \( O(N^2) \) operations. The FMM provides a way to reduce the number of operations to \( O(N) \). For the construction of the Fock matrix, one must determine the far-field potential. This section describes the FMM algorithm, emphasizing a new parallel algorithm that is suitable for the parallel building of the Fock matrix. To accomplish this it is necessary for the scalability of the algorithm to be less dependent on the number of computing nodes and the spatial distribution of data.

For brevity, only the 1-dimensional distribution of particles is illustrated in Figures 2–7. This discussion is readily extended to 3-dimensional charge distributions.

A common feature of hierarchical multipole methods is that the particles are recursively subdivided into a hierarchy of boxes or cells, based on their spatial positions. This hierarchy establishes a treelike data structure with a complete representation of the particles at each tier or level of subdivision. The algorithm begins by dividing a box containing all distributions (parent, root) into eight equal boxes (children). Recursive subdivision of this type leads to an “oct-tree” data structure. Each tier of the tree represents the entire set of particles, but it involves boxes of increasing spatial resolution formed by subdivision of their parents. Subdivision continues until the derived optimization criteria are satisfied. Boxes that do not contain any charges are not added to the tree.

![FIGURE 2](image1)

**FIGURE 2.** A 1-dimensional illustration of step 1 of the FMM. The multipoles of subdivision level 3 are collected to their parents in level 2 using the \( y^2 \) translations.

![FIGURE 3](image2)

**FIGURE 3.** The traditional method of step 2 of the FMM. At subdivision levels 2 and 3, each box collects its own local Taylor expansions using the \( y^2z \) translations.

Reductions in computational complexity are achieved by approximating particle distributions within a cell in terms of multipole expansions in the lowest subdivision level using the two recursion relations presented in Appendix A. Of course, the use of truncated multipole expansions alone will not re-
duce the $O(N^2)$ computational complexity, because the number of multipoles is proportional to the number of distributions that constitute the density. The FMM achieves linear scaling by hierarchically translating local multipoles and converting them into far-field Taylor expansions of the other boxes. In other words, instead of computing pairwise interactions among multipoles, the FMM algorithm collects contributions hierarchically, as described below.

**STEP 1: OBTAIN MULTipoLE EXPANSIONS AT LOWEST TREE LEVEL AND UPWARD PASS ($y2y$)**

The FMM algorithm starts by dividing a parent box, which spans all space, into a set of smaller child boxes (eight in 3 dimensions or two in 1 dimension). It is customary to choose the smallest cube $\mathcal{I}$ that encloses all the particles as the parent box. The child boxes are further divided, creating a hierarchical tree of boxes (see Fig. 2). Subdividing the parent box $N_s$ times yields $2^{N_s}$ (1 dimension) or $8^{N_s}$ (3 dimensions) child boxes in the lowest subdivision level. The number of divisions ($N_s$) must be wisely chosen to achieve linear scaling. A general procedure to obtain the optimal $N_s$ is given in a later section.

- $N_s=0$
  - 1

- $N_s=1$
  - 1
  - 2

- $N_s=2$
  - 1
  - 2
  - 3
  - 4

- $N_s=3$
  - 1
  - 2
  - 3
  - 4
  - 5
  - 6
  - 7
  - 8

**FIGURE 5.** A sequence to compute $y2z$ translations of step 2. In this sequence the $y2z$ translations are performed for the first box. The process continues until it arrives at the fifth box.

**FIGURE 6.** A sequence to compute $y2z$ translations of step 2 for the current parallel algorithm. There are three unique types of translations, which are distinguished by the length of the translation in the current example. Each unique type is distributed to a different compute node such that each compute node performs its unique type of translations. So, node0 performs only two actual translations.

**FIGURE 7.** An illustration of step 3. Parents of level 2 translate their local Taylor expansion to their children of level 3 using $z2z$ translations.
Meanwhile, \( N_s = 3 \) is chosen to illustrate the FMM algorithm.

Although the parent box is subdivided \( N_s \) times, the corresponding particles are not yet assigned to the boxes in the lowest subdivision level. A heap sort algorithm\(^{24}\) is used to distribute the particles among the corresponding boxes. This procedure is an \( O(N \log_2 N) \) process, where \( N \) is the number of particles. Because the heap sort algorithm need not assign particles to all boxes, it is necessary to remove empty boxes in the subsequent processes to both improve the performance and reduce the memory overhead of the FMM procedure.

Once the particles that belong to each box in the lowest subdivision level are determined, the multipole moments (\( \tilde{y}_m^l \)) for each particle are evaluated and summed about the center of the box for all boxes in the lowest subdivision level, \( N_s \). Two useful recurrence relations to construct \( \tilde{y}_m^l \) are presented in Appendix A. This procedure is \( O(N) \). At this point, each box in the lowest subdivision level has its own \( \{\tilde{y}_m^l\} \) that represents the particles in the corresponding boxes. When \( N_s \geq 3 \), the \( \tilde{y}_m^l \) of the child boxes in the lowest subdivision level are translated up to their parent boxes along the hierarchical tree and summed using eq. (28), the \( y_2y \) translation (see Fig. 2). In the next set of upward translations, these parents become the children of their own ancestors and the process continues until it reaches the level \( N_s = 2 \). The process requires \( 2^{N_s} \) (1 dimension) or \( 8^{N_s} \) (3 dimensions) \( y_2y \) translations at each \( N_s \), if the empty boxes have been removed. Because the computational effort needed to perform this step is negligible compared to \( y_2z \) translations, in the current algorithm all nodes perform the same computations. As a consequence, all nodes have identical \( \tilde{y}_m^l \) and subdivided boxes. In fact, \( \tilde{y}_m^l \) can also be distributed among the compute nodes. More details regarding memory overhead are discussed in Appendix B.

**STEP 2: \( y_2z \) TRANSLATION BASED ON INTERACTION LIST—CONVERTING MULTIPOLE MOMENTS INTO LOCAL TAYLOR EXPANSIONS**

Before presenting the algorithm for step 2, it is useful to discuss the FMM computational bottleneck. The majority of the computations required by the FMM occur in the translations of the spherical harmonics described by eqs. (28), (29), and (30). In particular, as explained below, 98% of the FMM overhead comes from the evaluation of the many \( y_2z \) translations in eq. (29). Therefore, it is critical to minimize the number of \( y_2z \) translations and parallelize the computation of these translations. Earlier parallel implementations\(^{3-5}\) relied on dividing the boxes of the tree among available processors, a natural way to tackle the problem. The main drawback of that approach is that because the data on different compute nodes are not independent of each other, there are significant internode communications. The amount of communication increases almost exponentially with the number of processors. Thus, the more the boxes of the tree are divided, the more internode communications there are. This has a serious impact on the scalability of the algorithm and limits the utility of parallel runs. Furthermore, this method can suffer from load-balancing problems if, as is likely, the particles are not uniformly distributed among the boxes. The strategy used in the current algorithm is to distribute the work rather than the data. As explained in Appendix B, the replicated memory can be minimized to the point where the actual FMM memory overhead is much smaller than that required for the quantum mechanical part.

In step 2 the \( \tilde{y}_m^l \) of the boxes in the “interaction list” are translated, converted to the \( z_m^l \) of a given box, and summed (see Fig. 3) according to eq. (29). The interaction list is uniquely defined at each subdivision level as explained below. The key here is to optimize the convergence of the infinite sum in eq. (29). To achieve convergence, \( \tilde{r}_c \) is the lesser and \( \tilde{r}_s \) is the greater of two arbitrary vectors: \( \tilde{r}_c \) corresponds to the relative positions of two charges with respect to the centers of their boxes, and \( \tilde{r}_s \) gives the relative position of the center of a neighbor box with respect to the center of the reference box. Thus, the norm of \( \tilde{r}_s \) is the length of the translation and the norm of \( \tilde{r}_c \) is the distance between a particle and the center of its box. To satisfy \( |\tilde{r}_s| > |\tilde{r}_c| \) and to accelerate convergence, first neighbors or even next nearest neighbors should not be in the interaction list. The well-separateness \( (ws) \) determines the neighbors that are not included in the interaction list. For instance, if \( ws = 1 \), the nearest neighbors are not included. The larger \( ws \) is, the faster the convergence. The greater the difference \( |\tilde{r}_s| - |\tilde{r}_c| \) is, the faster is the convergence, because then the FMM becomes a more accurate representation of the Coulomb interaction. However, increasing \( ws \) also increases the number of \( y_2z \) translations; this reduces the overall performance. Therefore, a proper choice of \( ws \) is necessary. The process continues until each box collects its \( z_m^l \). One can understand the number of translations needed with reference to Figures 2 and 3. If \( ws = 1 \), the interaction list for the second box from the left at the \( N_s = 2 \) level includes

---

The content is extracted from a scientific paper discussing the parallelization of the Fast Multipole Method (FMM) for computational chemistry. It details the steps involved in translating multipole moments and emphasizes the importance of load balancing and memory management to improve the performance of the algorithm.
the fourth box. At the \( N_s = 3 \) level the children of the second box \( (N_i = 2) \) are boxes 3 and 4. These two children inherit the contents of their parent box (e.g., the \( yz \) from box 4, \( N_i = 2 \)). Now, consider box 4 at \( N_s = 3 \). The interaction list of this box includes boxes 1, 2, and 6 (\( N_i = 3 \)). Boxes 3 and 5 are excluded, because they are nearest neighbors to box 4; boxes 7 and 8 are excluded, because they are already included at the \( N_i = 2 \) level. In general, the number of boxes in the interaction list is \( 2(2ws + 1) - 2ws + 1 \) in 1 dimension. In 3 dimensions this becomes \( 2(2ws + 1)^3 - (2ws + 1)^3 \). For \( ws = 1 \), this is \( 6^3 - 3^3 = 189 \); for \( ws = 2 \), it is \( 10^3 - 5^3 = 875 \) \( yz \) translations, corresponding to the number of boxes in the interaction list for a given box. In other implementations, two-way \( yz \) translations of each pair of boxes are required as illustrated in Figure 3. The total number of \( yz \) translations for the \( N_s \) level is then \( 2 \times 875 \times 8^{N_i} \) (\( ws = 2 \), 3D case). Therefore, this is the rate-determining step. Such a procedure yields doubly counted interactions between boxes.

If only the total interaction energy is needed, the algorithm can be modified so that only one-sided translations are allowed during this step, reducing the total number of \( yz \) translations to \( 875 \times 8^{N_i} \) (\( ws = 2 \)) as illustrated in Figure 4. The sequence of translations is reordered to optimize parallelization strategies. Traditionally, as illustrated in Figure 5, the procedure takes one box at a time and collects all the necessary translations in the interaction list. All the necessary \( yz \) translations are shown in Figure 5 for \( ws = 2 \) and \( N_s = 3 \). If one uses one-sided translations, only 10 (rather than 20) \( yz \) translations are needed. The procedure continues until it reaches the last box.

To optimize the scalability of the algorithm, the current procedure takes one type of translation at a time rather than one box at a time, where a translation type is determined by the length of the translation, that is, the number of boxes by which the two ends are separated. One then performs the translation between pairs of boxes that have the selected translation type as shown in Figure 6. The translation type is uniquely determined by the translation vector, the difference vector between the centers of two boxes. There are three unique translation types when \( ws = 2 \) and \( N_s = 3 \) in the 1-dimensional case, differing in the translation distances. The procedure continues until all types of translations are completed. There are 875 types of translations when \( ws = 2 \) for 3 dimensions. For the construction of the Fock matrix, each interaction must be double counted, because each element of the Coulomb component of the Fock matrix needs to feel the full potential due to the entire system.

There are many advantages to this approach. Parallelization becomes straightforward with this procedure, because it is easy to distribute the different types of translations to different computing nodes as illustrated in Figure 6. Consequently, each node collects some portion of \( z_{i1}^n \). This process does not require any internode communication. One drawback of this method is that each type of translation can have a different number of occurrences. In the example in Figure 6 the first translation type has two occurrences and the other two types have four occurrences. Thus, a load-balancing problem may occur. However, the next section illustrates that this is not a serious problem. Another advantage of the present approach is that, because all translations of a given type share the same rotation matrix \( D \), as well as the coefficients in eq. (29), \( D \) can be calculated once and used for \( 8^{N_i} \) translations. This removes a significant number of redundant computations.

**STEP 3: DOWNWARD PASS. z2z TRANSLATIONS**

The \( z2z \) translations as illustrated in Figure 7 are performed on all nodes. Because of the apportioning of the types of translations in step 2 to take advantage of parallelism, each node now has different \( z_{i1}^n \) accumulated in the boxes. The procedure continues until it reaches the lowest subdivision level. This step also does not require any internode communications. This process requires \( 2^{N_i} \) and \( 8^{N_i} \) \( z2z \) translations in 1 and 3 dimensions, respectively.

**STEP 4: EVALUATION OF FAR-FIELD AND NEAR-FIELD CONTRIBUTIONS**

Each node computes eq. (33) for each box in the lowest tree level to obtain the far-field contribution of the \( i \)th node \( (E_{far}^i) \) due to the particles that are more than \( ws \) boxes away from a given box. The near-field contributions of the \( i \)th node \( (E_{near}^i) \), which contain the interactions between particles in a given box and its near neighbor boxes, are evaluated from the exact Coulomb interaction, which can be easily parallelized. At this point, the \( i \)th node has its own \( E^i = E_{far}^i + E_{near}^i \). The total interaction energy \( (= \sum_i E^i) \) can be readily calculated by the standard function call of a message passing library.
ERROR ESTIMATION

Using eq. (33), the basic error estimate ($\epsilon$) of the FMM is

$$\epsilon = |\Phi_{\text{MM}} - \Phi_{\text{exact}}| = \left| \sum_{l=0}^{l_{\text{max}}} \sum_{m=-l}^{l} \frac{l! m!}{(l+m)!} |q_a q_b r_{<}^{l+m} - |q_a q_b r_{>}^{l+m} - \frac{q_a q_b}{r_{<} - r_{>}^l} | \right|. \quad (34a)$$

Because the second term on the right-hand side of eq. (34a) can be represented as an infinite Laplace expansion and the first term can be expressed as Legendre polynomials, eq. (34a) becomes

$$\epsilon = \left| \sum_{l=0}^{l_{\text{max}}} \frac{q_a q_b r_{<}^{l+m}}{l_{\text{max}} + 1} P_l(\cos \alpha) \right| \leq \left| \sum_{l=0}^{l_{\text{max}}} \frac{q_a q_b r_{<}^{l+m}}{l_{\text{max}} + 1} \right|. \quad (34b)$$

Greengard and Rokhlin\(^1\) adopted the inequality of $|P_l(\cos \alpha)| \leq 1$ in order to simplify the derivation and obtain an upper bound for the error estimation, where $\alpha$ is the angle between the two vectors $\vec{r}_{<}$ and $\vec{r}_{>}$. By neglecting the Legendre polynomial one obtains

$$\epsilon = \left| \frac{|q_a q_b|}{r_{<}^{l_{\text{max}} + 1}} \sum_{l=0}^{l_{\text{max}} + 1} \frac{q_a q_b r_{<}^{l+m}}{l_{\text{max}} + 1} \right|. \quad (34c)$$

$$\epsilon = \left| \frac{|q_a q_b|}{(ws + 1 - \sqrt{3}/2) d} \left( \frac{\sqrt{3}}{2 \cdot (ws + 1)} \right)^{l_{\text{max}} + 1} \right|. \quad (34d)$$

where $r_{<}$ is the maximum radius of the positions of particles with respect to the center of the box and $l_{\text{max}}$ is the highest multipole order of the expansion. In 3 dimensions, $r_{<}$ is $\sqrt{3}/2d$, where $d$ is the length of a side of a box as illustrated in Figure 8. In general, points may be anywhere in a given box. However, by assigning $r_{<}$ to the distance between the center and the corner of a box, one obtains an upper bound for the error estimate, because the error is determined by the ratio of $r_{<}$ and $r_{>}$. Recall that $r_{>} = (ws + 1)d$ is the minimum distance between the centers of two boxes in the interaction list. Thus, the error estimate depends on the dimension of the boxes. Suppose $\Xi$ is the smallest possible cube that encloses all the particles. If one increases the volume of $\Xi$, $d$ is increased accordingly. Since $\epsilon$ is inversely proportional to $d$, the error will decrease.

**FIGURE 8.** A 1-dimensional illustration of geometric parameters for eq. (34). $d$ is the side length of boxes in the lowest subdivision level. $ws$ is the well-separatedness, which determines $r_{>}$ of eq. (34), where $r_{>}$ is the separation between the two centers of boxes and $r_{<}$ is the maximum radius of the particle distribution.

**VFMM EVALUATION**

Petersen et al.\(^9\) showed that an improvement over existing FMM algorithms is to use lower values of $l$, $l_{\text{min}}$, for distant cells in the interaction list than for closer cells. In the present approach the formula for $l_{\text{min}}$ may be simplified, because the actual translations are performed along the $z$ axis:

$$l_{\text{min}} = \log \frac{\epsilon (n - \sqrt{3}/2)d}{|d|} / \log \frac{\sqrt{3}}{2n} - 1 \quad (35)$$

where $n + 1$ is the number of boxes between the two centers and $d$ is the length of a side of a box. The original algorithm\(^9\) required a sophisticated method to screen out pairs of translations. Because of the simplicity of eq. (35), this is not necessary in the current implementation.

**CHOOSING THE OPTIMUM SET OF PARAMETERS: OPS PROCEDURE**

As discussed in earlier sections, one has to determine $\epsilon$, $d$, $ws$, $l_{\text{max}}$, and $N_s$ to perform FMM computations. Choosing the optimum set of these parameters is not an easy task, because they are interrelated. White and Head-Gordon\(^1\)\(^b\) achieved a significant performance improvement using their fractional tier method. Their basic idea is that one can balance the near-field and far-field computational effort by minimizing the variation in the number of particles per lowest level box relative to the optimal value. However, the best choice for the optimum number of particles is not obvious, because one does not have control over numerical accuracy.

In this section, a stable and efficient method for determining the optimal parameters is proposed that guarantees both accuracy and performance. First, consider the relationships between the predicted errors based on eq. (34) and the actual calculated errors. The calculated error is defined as the
FIGURE 9. The expected error versus the actual error as a function of \( l_{\text{max}} \). There are 512 particles located at the corners of boxes in the \( N_s = 3 \) subdivision level. The expected errors are calculated by eq. (34), and the actual errors are the relative errors of the total Coulomb energy.

Calculated total energy relative to the exact energy at a given level of theory. The worst case, which would yield the largest error, occurs when the particles are at the corners of the boxes in the lowest level of a tree, in which case \( r_c \) becomes \((\sqrt{3}/2)d\). A test calculation was performed with 512 unit particles located at the corners of boxes in the \( N_s = 3 \) subdivision. The results are presented in Figure 9. The order of the multipole expansion \( (l_{\text{min}}) \) is determined parametrically by eq. (35). The actual errors are consistently lower than the predicted errors until the predicted errors decrease to \( 1 \times 10^{-6} \) and \( 1 \times 10^{-12} \) with \( w_s = 1 \) and 2, respectively. The corresponding typical \( l_{\text{max}} \) values are around 11 and 23 with \( w_s = 1 \) and 2, respectively. In the smaller error regions the total errors are dominated by the incomplete summations of eq. (29). Then a higher degree of multipole expansion does not improve the accuracy. In order to study possible degradation of the accuracy with a higher subdivision level, the same data set was used to calculate the errors with \( N_s = 4 \); the results are presented in Figure 10. The extra computations due to the use of a higher subdivision level do not change the accuracy significantly, and again the absolute lower error bound is determined by the incomplete summations in eq. (29).

The same test was performed on 40,000 randomly distributed unit charges, and the results are presented in Figure 11. Now the actual errors are much smaller than the predicted errors as compared with Figure 10, because the particles are no longer restricted to the corners of the boxes. The lower bounds are \( 1 \times 10^{-11} \) and \( 1 \times 10^{-12} \) with \( w_s = 1 \) and 2, respectively.

FIGURE 10. The expected error versus the actual error as a function of \( l_{\text{max}} \) with \( N_s = 3 \) and 4. There are 512 particles located at the corners of boxes in the \( N_s = 3 \) subdivision level. The expected errors are calculated by eq. (34), and the actual errors are the relative errors of the total Coulomb energy.

FIGURE 11. The expected error versus the actual error of 40,000 randomly distributed unit charges in the \( N_s = 3 \) subdivision level as a function of \( l_{\text{max}} \) with \( w_s = 1 \) and 2. The expected errors are calculated by eq. (34), and the actual errors are the relative errors of the total Coulomb energy.
and 2, respectively. Regardless of the number of data points, the absolute lower bound error of $1 \times 10^{-12}$ appears to consistently limit the accuracy for $w_s = 2$. This accuracy may be sufficient for most high accuracy quantum computations. For less accurate computations, where a tolerable error is greater than $1 \times 10^{-6}$, $w_s = 1$ may be adequate. The above studies show that $w_s = 2$ is necessary for high accuracy computations, and the smallest actual error one can obtain is about $1 \times 10^{-12}$. Therefore, the $w_s$ parameter needs to be determined by the magnitude of the acceptable errors, $\varepsilon$, to insure that the actual error is smaller than the predicted one.

For a given acceptable error $\varepsilon$ that determines $w_s$ and $l_{\text{max}}$, the next task is to determine the optimum values for the parameters $d$ and $N_s$ to achieve optimal performance. Petersen et al.9 deduced an explicit form for the computational overhead in a FMM calculation. Because the present FMM implementation is based on rotations, new sets of equations are necessary.

The “direct interaction” overhead ($T_d$) is obtained from the time required to calculate all pair interactions between particles that at the highest level of subdivisions reside in the same cell or in first neighbor cells ($w_s = 1$) or in first and second neighbor cells ($w_s = 2$). When the particles are randomly distributed, the number of particles in a box is $m_b = M/8N_s$, where $M$ is the total number of particles. Let us define $t_d$ as the time required to compute one pair interaction. Then the time required to compute the potential within each box is $\sim (m_b^2/2)t_d$, and the time required to compute the interactions between the current box and its neighbors is $\sim (m_b^2/2)t_dN_{bbr}$, where $N_{bbr} = (2w_s + 1)^3$ is the number of neighbor boxes. Because there are $8N_s$ boxes, the total $T_d$ for the 3-dimensional case may be written as

$$T_d = \frac{(2w_s + 1)^3M^2}{2 \cdot 8N_s}t_d.$$  \hfill (36)

Note that this estimate needs to be changed for Fock matrix construction, because in quantum computations, multilcenter two electron integrals are evaluated, not simply Coulomb interactions between particles, for the near field.

In steps 1 and 3, which involve children to parents and parents to children translations, to perform one general translation, each box requires two rotations and one translation. Because rotation and translation overhead require $(2l_{\text{max}} + 1)^3$ and $(l_{\text{max}} + 1)^3$ operations, respectively, the total overhead for one general translation is

$$T_{\text{tl}} = (2l_{\text{max}} + 1)^3 + (l_{\text{max}} + 1)^3)t_t.$$  \hfill (37)

The “transfer overhead” ($T_t$) can thus be written as

$$T_t = \sum_{n = 1}^{N_s} 8^n(2T_{t1} + T_{t2})$$

$$= \sum_{n = 1}^{N_s} 8^n\left\{[2(2w_s + 1)]^3 - (2w_s + 1)^3 + 2 \right\}$$

$$\cdot \left[2l_{\text{max}}^3 + (l_{\text{max}} + 1)^3\right]t_t$$

$$\approx \frac{8}{8 - 1}8N_s\left\{[2(2w_s + 1)]^3 - (2w_s + 1)^3 + 2 \right\}$$

$$\cdot \left[2l_{\text{max}}^3 + (l_{\text{max}} + 1)^3\right]t_t.$$ \hfill (39)

The total computational overhead ($T$) is $T_t + T_d$. For convenience, $8N_s$ is denoted as $\alpha$. The optimum number of boxes in the highest subdivision level ($a_0$) is determined by the partial derivative of $T$ with respect to $a$: $a_0 = M\left(\frac{7}{2} \cdot \frac{8}{7}(2w_s + 1)^3\right)$

$$\times \left\{\left[(2w_s + 1)^3 - (2w_s + 1)^3 + 2 \right]\right\}^{-1}$$

$$\cdot \left[2l_{\text{max}}^3 + (l_{\text{max}} + 1)^3\right]1/2.$$ \hfill (40)

As mentioned above, $w_s$ and $l_{\text{max}}$ are parametrically determined by $\varepsilon$. If $\alpha = a_0$, $T$ becomes the optimum $T_0$, that is

$$T_0 = \left(\frac{2}{7} \cdot \frac{8}{7}(2w_s + 1)^3\right)$$

$$\cdot \left\{\left[(2w_s + 1)^3 - (2w_s + 1)^3 + 2 \right]\right\}^{-1}$$

$$\cdot \left[2l_{\text{max}}^3 + (l_{\text{max}} + 1)^3\right]1/2.$$ \hfill (41)

Once $w_s$, $l_{\text{max}}$, $t_t$, and $t_d$ are determined, the total computational overhead is proportional to $M$. So, with $T_0$ chosen in this manner, the FMM is linear.
with respect to \( M \). The multiplying factor in eq. (41) determines the slope of linear scaling. Once \( \omega \) is set, this factor is proportional to \( l_{\text{max}} \), which is a typical scale factor for rotation-based FMM. Therefore, in general, accurate calculations yield a larger slope.

Now a procedure is needed that can produce the optimum \( a_0 \) for a given \( \varepsilon \) in the calculation. According to the OPS procedure, the volume of \( \mathcal{I} \) that encloses all the particles is adjusted such that subdividing \( \mathcal{I} \) only yields \( a_0 \) occupied boxes in the highest subdivision level \( N_s \). The smallest \( N_s \), which yields at least \( a_0 \) occupied boxes, can be obtained from

\[
N_s = \lceil \log_{a_0} \varepsilon \rceil \tag{42}
\]

where \( \lceil x \rceil \) yields the smallest integer that is greater than \( x \). To obtain \( \omega \) boxes in the highest subdivision level, one renormalizes the length \( (L_r) \) of the \( \mathcal{I} \) so that the volume of the cube is renormalized,

\[
L_r' = f \cdot L_c
\]

where \( f \) is given by

\[
f = \left( \frac{8^{N_s}}{a_0} \right)^{1/3} \tag{44}
\]

and the volume of the cube becomes \( L_r'^{3/2} \). This procedure can yield many empty boxes and only insures that approximately \( a_0 \) occupied boxes exist in the highest subdivision level. Note that once the cube containing all the particle locations is renormalized, \( d \), the length of the box at the highest subdivision level is changed. Consequently, \( l_{\text{max}} \) is changed for a given \( \varepsilon \) through eq. (34). Therefore, eq. (40) to obtain \( a_0 \) must be solved self-consistently among these parameters. Fortunately, since this iterative procedure converges very rapidly, only two iterations are usually necessary. To perform this procedure, one has to determine the ratio \( R = t/d_t \) by test computations. It is a reasonable assumption that \( R \) is nearly independent of the computational hardware.

The entire procedure is summarized in Figure 12. Initially, the user chooses the acceptable error \( \varepsilon \), which subsequently determines \( \omega \). The program determines an initial \( L_c \) on the basis of the distribution of data and sets the initial \( N_s \) to 3. The program then calculates \( d \), \( l_{\text{max}} \), and an initial \( a_0 \). Based on these parameters, the program then calculates a new \( N_s \) using eq. (42), \( f \) using eq. (44), and \( L_r' \) using eq. (43). \( L_r' \) yields a new \( d \) and subsequently a new \( l_{\text{max}} \) using eq. (34). The new \( l_{\text{max}} \) determines a new \( a_0 \) through eq. (40). The process continues until \( a_0 \) reaches self-consistency.

The OPS procedure presented above may yield two sets of parameters in some ranges of \( M \). An example is shown for \( M = 120,000 \) in Table I. As the iteration progresses, two alternating sets of parameters are found. However, as verified in representative test cases, these two sets of parameters are expected to exhibit identical performance.

In order to illustrate the usefulness of the OPS procedure, test calculations were performed as a function of the number of randomly distributed unit charges. Plots of time against the number of particles for the acceptable error levels \( 1 \times 10^{-6} \) and \( 1 \times 10^{-12} \) are presented in Figures 13 and 14, respectively. Corresponding values as obtained with the OPS are listed in Tables II and III. Note that the OPS procedure does not require explicit values of the parameters. The necessary FMM parameters are determined implicitly, depending on a user specified \( \varepsilon \) and \( M \). The calculations were done on an IBM 604 PPC 133-MHz single node workstation. In Figures 13 and 14 the thin solid lines represent \( O(N^2) \) scaling of the direct Coulomb equation, the dotted lines represent nonlinear scaling of FMM with a fixed \( N_s \) parameter, and the thick solid lines show

---

**FIGURE 12.** A flow chart to determine the optimal set of FMM parameters: \( \varepsilon \), acceptable error; \( \omega \), well-separateness; \( L_c \), initial side length of the root box; \( N_s \), subdivision level; \( d \), side length of the box in the lowest subdivision level; \( l_{\text{max}} \), highest order of multipole expansion; \( a_0 \), optimal number of boxes.
TABLE I.
Example Run of Optimum Parameter Searching (OPS) Procedure with $\varepsilon = 10^{-6}$ ($ws = 2$) and $M = 120,000$.

<table>
<thead>
<tr>
<th>Iteration</th>
<th>$l_{\text{max}}$</th>
<th>$\alpha_0$</th>
<th>$N_s$</th>
<th>$L_c$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial</td>
<td>11</td>
<td>2281.81287425100845</td>
<td>3</td>
<td>2.0</td>
</tr>
<tr>
<td>1</td>
<td>11</td>
<td>2281.81285020654332</td>
<td>4</td>
<td>2.430639687046498</td>
</tr>
<tr>
<td>2</td>
<td>12</td>
<td>2014.21259216605927</td>
<td>4</td>
<td>2.53383785262995165</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>2281.81285020654332</td>
<td>4</td>
<td>2.430639687046498</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>2014.21259216605927</td>
<td>4</td>
<td>2.53383785262995165</td>
</tr>
</tbody>
</table>

the linear scaling $O(N)$ that is obtained using the current procedure. Both figures illustrate that the procedure guarantees linear scaling with respect to $M$ with complete control over the acceptable error.

PARALLEL PERFORMANCE

Randomly distributed unit charges were used to test the parallel performance of the new algorithm. Timings with one and eight processors ($P$) are presented in Figure 15. An eight-processor 200-MHz IBM Power3 cluster was used. The plot clearly shows linear scaling with respect to the number of particles and parallelization does not degrade the linearity. The actual speedup on eight processors is 6.9. The degradation of parallel speedup is due to the nonuniform number of actual translation operations of different types of translations in step 2 and nonparallel steps in the algorithm. As discussed earlier, steps 1 and 3 are not parallelized because of the negligible overhead compared to step 2 (load balancing). As illustrated in Figure 6, each translation type has its own number of occurrences. However, this irregularity can be partially remedied by using a statistically averaged number of actual translations. The merit of our parallel implementation is that one can in principle utilize up to 875 processors for $ws = 3$ with little overhead, because there is virtually no internode communication.

FIGURE 13. FMM timings against the direct method at tolerable error $= 1 \times 10^{-6}$. The thin solid line represents $O(N^2)$ scaling of the direct Coulomb equation. The dotted lines represent nonlinear scaling with fixed $N_s$ parameters. The thick solid line represents the renormalization method that exhibits true linear scaling with respect to the number of particles. The $N_s$ of the renormalization method is determined implicitly, depending mostly on user specified error and $M$. Thus, $N_s$ is not a fixed number.

FIGURE 14. FMM timings against the direct method at tolerable error $= 1 \times 10^{-12}$. The thin solid line represents $O(N^2)$ scaling of the direct Coulomb equation. The dotted lines represent nonlinear scaling with fixed $N_s$ parameters. The thick solid line represents the renormalization method that exhibits true linear scaling with respect to the number of particles. The $N_s$ of the renormalization method is determined implicitly, depending mostly on user specified error and $M$, so $N_s$ is not a fixed number.
TABLE II. $\alpha_0$, $N_s$, and $l_{\text{max}}$ as Obtained with Optimum Parameter Searching (OPS) Procedure with $\varepsilon = 10^{-6}$ ($ws = 2$).

<table>
<thead>
<tr>
<th>$M$</th>
<th>$\alpha_0$</th>
<th>$N_s$</th>
<th>$l_{\text{max}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3000</td>
<td>570.453212551635829</td>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td>6000</td>
<td>1140.90642510327166</td>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td>9000</td>
<td>1711.35963765490737</td>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td>12,000</td>
<td>2281.81285020654332</td>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td>15,000</td>
<td>2517.76574020757380</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>18,000</td>
<td>3021.3188824908856</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>21,000</td>
<td>3524.8720369060377</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>24,000</td>
<td>4028.4251843211853</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>27,000</td>
<td>4531.97833237363284</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>30,000</td>
<td>5035.5314804151476</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>33,000</td>
<td>5539.08462845666236</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>36,000</td>
<td>6042.6377649817712</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>39,000</td>
<td>6546.19092453969279</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>42,000</td>
<td>7049.74407258120755</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>45,000</td>
<td>7553.2972206272231</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>48,000</td>
<td>8056.8503666423707</td>
<td>5</td>
<td>12</td>
</tr>
</tbody>
</table>

MEMORY REQUIREMENTS

The total memory requirements for a single processor, including the space for the coordinates and charges of particles, with respect to the number of particles are presented in Figure 16. The plot shows that memory requirements also grow linearly with $M$. This is accomplished by only saving the multipole expansions of occupied boxes in memory. Because the number of occupied boxes increases linearly with $M$, so does the memory requirement. The current parallel implementation is based on using a standard message passing library. We are in

TABLE III. $\alpha_0$, $N_s$, and $l_{\text{max}}$ as Obtained with Optimum Parameter Searching (OPS) Procedure with $\varepsilon = 10^{-12}$ ($ws = 2$).

<table>
<thead>
<tr>
<th>$M$</th>
<th>$\alpha_0$</th>
<th>$N_s$</th>
<th>$l_{\text{max}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3000</td>
<td>208.860725281115378</td>
<td>3</td>
<td>22</td>
</tr>
<tr>
<td>6000</td>
<td>417.721450562230757</td>
<td>3</td>
<td>22</td>
</tr>
<tr>
<td>9000</td>
<td>626.582175843346135</td>
<td>4</td>
<td>22</td>
</tr>
<tr>
<td>12,000</td>
<td>835.442901124461514</td>
<td>4</td>
<td>22</td>
</tr>
<tr>
<td>15,000</td>
<td>1044.30362640557701</td>
<td>4</td>
<td>22</td>
</tr>
<tr>
<td>18,000</td>
<td>1253.16435168692227</td>
<td>4</td>
<td>22</td>
</tr>
<tr>
<td>21,000</td>
<td>1369.82310628675054</td>
<td>4</td>
<td>23</td>
</tr>
<tr>
<td>24,000</td>
<td>1565.51217407057208</td>
<td>4</td>
<td>23</td>
</tr>
<tr>
<td>27,000</td>
<td>1761.20113665439362</td>
<td>4</td>
<td>23</td>
</tr>
<tr>
<td>30,000</td>
<td>1956.89015183821493</td>
<td>4</td>
<td>23</td>
</tr>
<tr>
<td>33,000</td>
<td>2152.57916702203647</td>
<td>4</td>
<td>23</td>
</tr>
<tr>
<td>36,000</td>
<td>2348.2681220585801</td>
<td>4</td>
<td>23</td>
</tr>
<tr>
<td>39,000</td>
<td>2543.9571938967955</td>
<td>4</td>
<td>23</td>
</tr>
<tr>
<td>42,000</td>
<td>2739.6421257350109</td>
<td>4</td>
<td>23</td>
</tr>
<tr>
<td>45,000</td>
<td>2935.3352277532263</td>
<td>4</td>
<td>23</td>
</tr>
<tr>
<td>48,000</td>
<td>3131.02424294114417</td>
<td>4</td>
<td>23</td>
</tr>
</tbody>
</table>

FIGURE 15. FMM timings with one and eight processors with the acceptable error $\varepsilon_{\text{Y}}^{\theta, \phi}$. The renormalization method exhibits true linear scaling with respect to the number of particles. Parallelization does not degrade the linearity of the current method. The parallel efficiency is about 7.

FIGURE 16. The total memory requirements (MB) of one processor with an acceptable error of $1 \times 10^{-6}$. The memory requirement increases linearly with the number of particles.
the process of applying the global array concept of DDI\textsuperscript{25} to this algorithm.

### Conclusions

It has been shown that the three key translation equations can be readily deduced from the generalized equations.\textsuperscript{20} Because two rotations and one translation are needed for a general translation with rotation-based FMM, it is preferable to simplify rotation formulas rather than the translation formula. Based on this idea, basic regular and irregular harmonics are redefined and yield a compact formalism. Furthermore, the current formalism is advantageous, because \((2L + 1)\) terms are required to perform a rotation, while less than \(L\) terms are needed to perform a translation along a specialized axis. The rotation-based FMM can be further simplified, yielding a very compact FMM formalism.

A new parallel FMM algorithm is introduced that does not require internode communication. This new method is suitable for the parallel construction of the Fock matrix of quantum calculations. Instead of assigning divided space to compute nodes, unique translation types are assigned to compute nodes for the parallelization of the FMM. The former approach suffers from rapidly growing internode communications as a function of space subdivision and from load-balancing problems. In contrast, the implementation described in the current work does not suffer from these problems, so it is expected to provide more robust performance. In addition, in principle our implementation can work with any number of processors, limited only by the scaling as the number of processors grows.

A stable and efficient method for determining the optimum set of FMM parameters (OPS) via a user specified acceptable error is established using a self-consistent process to ensure both accuracy and optimal performance. Because the computational overhead depends most strongly on the number of boxes, the iteration converges very rapidly. The new self-consistent procedure achieves linear scaling with respect to the number of particles with complete control over the actual error.

In the current implementation, the memory requirement also grows linearly with the number of particles. This is achieved by saving only the multipoles of occupied boxes.

We are in the process of applying our new FMM algorithm to the construction of the Fock matrix.

### Appendix A: Recurrence Relations for \(\tilde{y}_l^m\)

Although associate Legendre functions satisfy numerous recurrence relations, the following recurrence on \(l\) is numerically stable:

\[
(l - m)P_l^m = x(2l - 1)P_{l-1}^m - (l + m - 1)P_{l-2}^m \quad (A.1)
\]

where \(x = \cos \theta\).

Multiplying both sides by

\[
(-1)^m r \left[ \frac{(l - m)!}{(l + m)!} \right]^{1/2} e^{-i \phi},
\]

one finds the following recurrence relation among \(\tilde{y}_l^m\):

\[
\tilde{y}_l^m = \frac{r}{[(l - m)(l + m)]^{1/2}} \left\{ x(2l - 1)\tilde{y}_{l-1}^m - r[(l - m - 1)(l + m - 1)]^{1/2}\tilde{y}_{l-2}^m \right\}. \quad (A.2)
\]

There is a useful closed-form expression when \(l = m\):

\[
P_m^m = (2m - 1)!(1 - x^2)^{m/2}. \quad (A.3)
\]

Multiplying both sides with

\[
(-1)^m r \left[ \frac{(l - m)!}{(l + m)!} \right]^{1/2} e^{-i \phi}
\]

produces an expression for \(\tilde{y}_m^m\):

\[
\tilde{y}_m^m = (-1)^m \left[ \frac{\rho_m^m}{(2m)!^{1/2}(2m - 1)!!(1 - x^2)^{m/2}} \right] e^{-i \phi}. \quad (A.4)
\]

From (A.4) one can readily obtain another recurrence relation:

\[
\tilde{y}_m^m = -r \left[ \frac{(2m - 1)(1 - x^2)}{2m} \right]^{1/2} e^{-i \phi} \tilde{y}_{m-1}^{m-1} \quad (A.5)
\]

It is noted that \(\tilde{y}_0^0 = q\).

### Appendix B: Memory Overhead in Parallel Algorithm for Electronic Structure Calculations

The multipole expansion (the charge-weighted scaled regular harmonics) of a particle requires \((L + 1)^* (L + 2)^* 8\) bytes of memory, where \(L\) is the highest order of the expansion. For accurate quantum calculations, normally more than seven-digit accuracy to the right of the decimal, \(L\) needs to be larger than 20. So, in such a case, 3696 bytes are needed. In the case of a buckytube (392 carbon atoms) calculation with the STO-3G basis set
with 1960 basis functions, 2,291,366 expansions are needed, which require 8.5 gigabytes (GB) of memory \((M_E)\). At this system size the memory overhead of the quantum part is about 270 megabytes (MB) of memory \((M_Q)\). For common FMM quantum calculations with about 10,000 basis functions, the \(M_E\) is in the range of 100 GB. Clearly, this sets the limit of FMM applicability, even with a distributed data algorithm. The next largest overhead arises from the multipole of each box. In the example we have 1477 nonempty boxes (including all levels of subdivision and \(N_s = 4\)) requiring about 11 MB of memory \((M_R)\) for both regular and irregular harmonics. This is much smaller than \(M_E\) and \(M_Q\). Thus, clearly \(M_E\) is the real memory overhead.

One approach to reducing the value of \(M_E\) is the following. Because of the unique features of quantum calculations, if we evaluate the multipole at the center of a distribution, the highest multipole order of a Gaussian basis product distribution is limited to \(l^1 + l^2\), where \(l^1\) and \(l^2\) are the angular momenta of basis functions 1 and 2 of the distribution.\(^{26}\) So if they are 2s or 2p orbitals, the highest nonzero multipole order is 4, which requires only 240 bytes of memory, reducing the \(M_E\) to 550 MB \((M_{RE})\). Of course, since the real origin of such an expansion does not correspond to the center of a distribution, in general, it needs to be translated to the real origin in the calculation. Consequently, the \(M_{RE}\) adds additional workload while yielding an order of magnitude smaller memory overhead. Because the most time consuming part of the calculation is still the real integral evaluations, the translations do not add significantly to the total workload and provide good parallelism. This algorithm is implemented to the \textit{ab initio} program GAMESS\(^{27}\) and will be distributed for public use.

So, depending on the available hardware, one can choose an appropriate method. In our implementations\(^{28}\) we distribute \(M_E\) among available processors. Depending on the case, we distribute the \(M_E\) or the \(M_{RE}\) or save the distributed \(M_E/M_{RE}\) to the local hard disk. This procedure virtually removes \(M_E\) overhead and requires one global sum of regular harmonics of each box.

Therefore, the only replicated memory in our implementation is \(M_E\), which is quite manageable with normal workstations. Although this implementation may not ultimately be the best solution, it overcomes the obstacles of distributed data algorithms without oppressive memory overhead.

---

**Acknowledgments**

C.H.C. and M.S.G. acknowledge support by a DoD CHSSI grant and a grant from the Air Force Office of Scientific Research. K.R. acknowledges support by the Division of Chemical Sciences, Office of Basic Energy Science, U.S. Department of Energy. The authors are grateful to Dr. Michael Schmidt and Prof. Martin Head-Gordon for valuable discussions during the course of this work.

---

**References**

20. Steinborn, E. O.; Ruedenberg, K. Adv Quantum Chem 1973, 7, 1, and see eqs. (307), (309), (315), and (348).