SIAM REVIEW
Vol. 43, No. 2, pp. 289–300

A Matrix Version of the Fast Multipole Method*

Xiaobai Sun†
Nikos P. Pitsianis†‡

Abstract. We present a matrix interpretation of the three-dimensional fast multipole method (FMM). The FMM is for efficient computation of gravitational/electrostatic potentials and fields. It has found various applications and inspired the design of many efficient algorithms. The one-dimensional FMM is well interpreted in terms of matrix computations. The three-dimensional matrix version reveals the underlying matrix structures and computational techniques used in FMM. It also provides a unified view of algorithm variants as well as existing and emerging implementations of the FMM.

Key words. fast multipole method, matrix representation

AMS subject classifications. 40C05, 33F05, 68W25

PII. S0036144500370835

1. Introduction.

The fast multipole method (FMM) introduced by Greengard and Rokhlin [13] is for efficient computation of the dense matrix-vector product $p = Aq$ arising in the computation of gravitational or electrostatic potentials and fields. Matrix A can be simply described by its elements

$$A(i,j) = \frac{1}{\|t_i - s_j\|_2}, \quad t_i,\ s_j \in \mathcal{R}^3, \quad i = 1:m, \quad j = 1:n, \tag{1.1}$$

where m and n are large. We call $f(t,s) = 1/\|t-s\|_2$ the influence or response function of a charge at the *source point* s upon the potential at the *target point* t. The potential at the target point t_i due to the charges q_j at n source points s_j, $j = 1:n$, is $p(i) = \sum_{j=1:n} A(i,j)q_j$. The target point may also be called the evaluation point or the observation point in different application problems. Direct evaluation of the potentials at m target points due to n source charges requires $O(mn)$ arithmetic operations.

In the ideal case where s_i and t_j are the nodes on a tensor product grid in $\mathcal{R}^3$, matrix A is block Toeplitz at three block levels in appropriate orderings, each level corresponding to a dimension in the Cartesian coordinate system. We may simply say that A is Toeplitz in three dimensions. Via the use of a fast Fourier transform (FFT) algorithm [22], the arithmetic complexity of a matrix-vector product with A of order n is $O(n \log n)$. In many computational problems, the spatial distribution of

*Received by the editors April 12, 2000; accepted for publication (in revised form) November 15, 2000; published electronically May 2, 2001 This work has been supported in part by DARPA/DSO grant DABT63-98-1-0001 and NSF/CISE grant CDA-9726370.
http://www.siam.org/journals/sirev/43-2/37083.html

†Department of Computer Science, Duke University, Durham, NC 27708 (xiaobai@cs.duke.edu, nikos@cs.duke.edu).
‡BOPS Inc., Chapel Hill, NC 27514.

source/target points is not ideal and the matrix no longer has the Toeplitz structure in any ordering. Embedding unevenly distributed points in a tensor product grid in order to use an FFT algorithm may increase the size of the matrix significantly. In contrast, the FMM permits various distributions of source/target points and has complexity $O(n)$.

The FMM has many applications in computational physics, chemistry, engineering, and applied mathematics. FMM ideas have inspired many other efficient algorithms for large-scale matrix computations; see, for instance, [4, 5, 7, 11, 12, 18, 20].

The matrix interpretation of one-dimensional FMMs of complexity $O(n \log n)$ is well presented in [14] and has facilitated an understanding of FMM ideas. Two- and three-dimensional FMMs are much more important in practice and have given rise to various implementations. We present in this paper a matrix interpretation of three-dimensional FMMs of complexity $O(n)$ as well as $O(n \log n)$. The matrix version is not merely a replacement for multiple summations, recursions, repetitions, and indices by simple and clean matrix notation; it reveals and highlights the underlying matrix structures exploited by FMMs and encapsulates the details properly and systematically. The matrix version influences application, generalization, and implementation aspects of the FMM. The application and generalization of the FMM has been the work of computational experts. The matrix viewpoint may help make the three-dimensional FMM understandable to more computational scientists. There exist numerous implementations of the FMM [3, 17, 19, 21, 23] and there are still more implementations likely to emerge. Analogous to the discrete Fourier transform (DFT) matrix factorization interpretation of FFT algorithms, the matrix interpretation of the FMM provides a systematic way to bridge various computation implementations and the underlying mathematics and numerics. It helps identify possible off-line computations and possible computation sequences in order to reduce repeated computations or circumvent numerical difficulties due to finite-precision arithmetic. Other important implementation issues include vectorization, memory reference locality, and parallelization.

The following notation is used throughout this paper. Matrices and vectors are denoted by upper case and lower case letters, respectively. The colon notation, such as $-p : p$, specifies an index enumeration, as in Fortran 90 and MATLAB. When the colon notation is used as a subscript of a matrix, it refers to the whole range of rows and/or columns. Matrices and vectors are often given by enumerating their elements within a pair of square brackets. The function $\mathrm{diag}(\cdot)$ is used to denote a diagonal or block diagonal matrix. The symbol $\otimes$ stands for the Kronecker product of two matrices and $\odot$ for the Hadamard (elementwise) product of two matrices.

The rest of this paper is organized as follows. In section 2, we present a matrix factorization, derived from an elementwise expansion, when the sources are well separated from the targets. In section 3, we introduce the FMM partition-and-split scheme to reveal and exploit the matrix structure, and we describe two categories of efficient algorithms of complexity $O(n \log n)$. In section 4, we describe algorithms of complexity $O(n)$. These algorithms are asymptotically optimal, up to a constant factor. Section 5 concludes the paper.

2. Elementwise Expansion and Blockwise Factorization. Elementwise expansions and matrix factorizations always go hand in hand in matrix computations. For example, let $A = U\Sigma V^{\mathrm{H}}$ be the singular value decomposition of A. Then, every element $A(i, j)$ is expressed in an expanded form $A(i, j) = U(i, :)\Sigma V(j, :)^{\mathrm{H}}$. The first novel idea of the FMM is to establish a matrix factorization from an analytic

elementwise expansion.

Let s_c and δs be the center and radius of the source set in the sense that $\|s_j - s_c\|_2 \le \delta s$. Let t_c and δt be the center and radius of the target set.

THEOREM 2.1. *Let the matrix of* (1.1) *be defined on a source set centered at* s_c *with radius* δs *and a target set centered at* t_c *with radius* δt. *Assume that* $t_c \neq s_c$ *and for some* $\alpha < 1$,

$$\delta t + \delta s = \alpha \|t_c - s_c\|_2. \tag{2.1}$$

Then, for any integer $p \ge 0$,

$$A = A_r + A \odot E,$$

where matrix A_r *is of the factored form*

$$A_r = V_t^{\mathrm{H}} T_{t_c,s_c} V_s \quad \textit{with} \quad T_{t_c,s_c} \in \mathcal{C}^{r\times r}, \quad r = (p+1)(2p+1), \tag{2.2}$$

and the elements of E *are bounded as follows:*

$$|E(i,j)| \le \frac{1+\alpha}{1-\alpha}\alpha^{p+1}. \tag{2.3}$$

We make the following comments before presenting the proof.

- The theorem requires that sources and targets be well separated in the sense of (2.1). The parameter α bounds from below the distance of any point in the source set from any point in the target set, relative to the distance between the two centers. Such matrix may be a subblock of a general matrix (1.1). We call α the *separation ratio* for the matrix (block).
- The approximation accuracy of A_r to A is described in terms of elementwise relative error.
- The approximation can be made in arbitrarily specified accuracy by adjusting the parameters p and α. We call p the *expansion order*. The relative error decreases as p increases or as α decreases.
- The rank of A_r is no greater than r, the order of the matrix factor T_{t_c,s_c}, which is determined by p. The elements of T_{t_c,s_c} depend only on the distance between the two centers. In other words, T_{t_c,s_c} is independent of m and n. When m and n are larger than r, A_r is a lower rank matrix.
- Each factor of A_r can be formed and saved in a compact form. The matrix-vector product with A_r in the factored form takes at most $2r(r+m+n)$ arithmetic operations.

Proof. The approximate factorization in (2.2) is established from an elementwise approximation. To obtain an elementwise expansion, we use the source center and target center as the reference points and expand the element $1/\|t_i - s_j\|_2$ about $t_c - s_c$; see Figure 2.1.

First, we rewrite $t_i - s_j$ as two terms and represent the terms in spherical coordinates,

$$t_i - s_j = (t_c - s_c) - (\delta s_j - \delta t_i) = (\rho_c, \alpha_c, \beta_c) - (\rho_d, \alpha_d, \beta_d), \tag{2.4}$$

where $\delta t_i = t_i - t_c$ and $\delta s_j = s_j - s_c$. Applying the multipole expansion theorem [9] to the right-hand side of (2.4), we have

$$\begin{aligned} A(i,j) &= A_r(i,j) + \epsilon(i,j), \\ A_r(i,j) &= \sum_{n=0}^{p} \sum_{m=-n}^{n} \frac{Y_{n,m}(\alpha_c,\beta_c)}{\rho_c^{n+1}} \rho_d^n Y_{n,-m}(\alpha_d,\beta_d), \end{aligned} \tag{2.5}$$

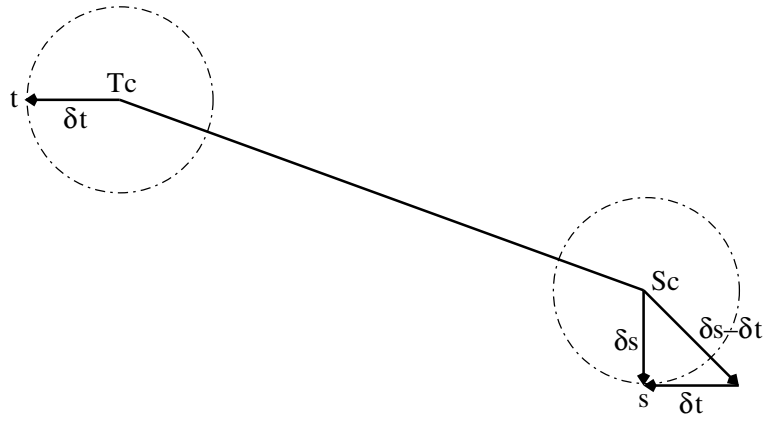

Fig. 2.1 *Expansion about $t_c - s_c$ and separation of δt and δs.*

where $Y_{n,m}(\cdot,\cdot)$ are spherical harmonic functions of degree n and order m and the approximation error in $A_r(i,j)$ is bounded as follows:

$$|\epsilon(i,j)| \leq \frac{1}{\rho_c} \sum_{n=p+1}^{\infty} \left(\frac{\rho_d}{\rho_c}\right)^n \leq \frac{1}{\rho_c} \frac{\alpha^{p+1}}{1-\alpha} \leq A(i,j) \frac{1+\alpha}{1-\alpha} \alpha^{p+1}.$$

Notice that the factors $Y_{n,m}(\alpha_c, \beta_c)/\rho_c^{n+1}$ in the expansion of $A_r(i,j)$ of (2.5) are common to all source-target pairs (t_i, s_j).

Next, we will decouple $\delta t_i = (d_t, \theta_t, \phi_t)$ and $\delta s_j = (d_s, \theta_s, \phi_s)$ into the factors $\rho_d^n Y_{n,m}(\alpha_d, \beta_d)$ in the expansion of $A_r(i,j)$. Define

$$Z_{l,k}(d,\theta,\phi) \stackrel{\text{def}}{=} \begin{cases} d^l A_{l,k} Y_{l,k}(\theta,\phi), & |k| \leq l, 0 \leq l \leq p, \\ 0, & \text{otherwise}, \end{cases}$$

$$A_{n,m} \stackrel{\text{def}}{=} \frac{(-1)^n (-1)^{|m|/2}}{\sqrt{(n-|m|)!(n+|m|)!}}.$$

By the third addition theorem of spherical harmonics (see [9] and the references therein),

$$A_{n,m} \rho_d^n Y_{n,-m}(\alpha_d, \beta_d) = \sum_{l=0}^{n} \sum_{k=-l}^{l} (-1)^l Z_{l,k}(\delta t_i) Z_{n-l,-m-k}(\delta s_j). \tag{2.6}$$

From (2.5) and (2.6) we have

$$A_r(i,j) = \sum_{n=0}^{p} \sum_{m=-n}^{n} \sum_{l=0}^{n} \sum_{k=-l}^{l} \frac{Y_{n,m}(\alpha_c, \beta_c)}{A_{n,m} \rho_c^{n+1}} (-1)^l Z_{l,k}(\delta t_i) Z_{n-l,-m-k}(\delta s_j).$$

With the embedding functions

$$X_{n,m}(\rho,\alpha,\beta) \stackrel{\text{def}}{=} \begin{cases} \dfrac{Y_{n,m}(\alpha,\beta)}{\rho^{n+1} A_{n,m}}, & |m| \leq n, \quad 0 \leq n \leq p, \\ 0, \quad \text{otherwise}, \end{cases}$$

we rearrange the summations in $A_r(i,j)$ and get

$$\begin{aligned} A_r(i,j) &= \sum_{l=0}^{p}\sum_{k=-p}^{p}\sum_{n=0}^{p}\sum_{m=-p}^{p} X_{n,m}(t_c - s_c)(-1)^l Z_{l,k}(\delta t_i) Z_{n-l,-m-k}(\delta s_j), \\ &= \sum_{l=0}^{p}\sum_{k=-p}^{p}\sum_{n=0}^{p}\sum_{m=-p}^{p} X_{n+l,-m-k}(t_c - s_c)(-1)^l Z_{l,k}(\delta t_i) Z_{n,m}(\delta s_j). \end{aligned}$$

Thus, in matrix-vector form,

$$A_r(i,j) = v(\delta t_i)^{\mathrm{T}} T_{t_c,s_c}\, v(\delta s_j), \tag{2.7}$$

where

$$v(w)^{\mathrm{T}} \stackrel{\mathrm{def}}{=} [Z_{0,-p:p}(w), \ldots, Z_{p,-p:p}(w)],$$

and matrix T_{t_c,s_c} is of order $(p+1)(2p+1)$ with elements defined as follows:

$$T_{t_c,s_c}(i,j) = (-1)^l X_{n+l,-m-k}(t_c - s_c), \qquad \begin{aligned} i &= l(2p+1) + (p+1+k), \\ j &= n(2p+1) + (p+1+m). \end{aligned}$$

The matrix factor T_{t_c,s_c} in (2.7) depends only on the distance between the centers. The approximation $A_r(i,j)$ in (2.7) is therefore bilinear in $v(\delta t_i)$ and $v(\delta s_j)$. Finally, aggregating the v-vectors over the points in each set,

$$V_s(:,j) = v(\delta s_j), \quad V_t(:,i) = v(\delta t_i),$$

gives the factorization of A_r in (2.2). □

We call $v(t_i - t_c)$ and $v(s_j - s_c)$ the *local expansion vectors* of s_j and t_i about their respective reference points s_c and t_c. They are separated and coupled by T_{t_c,s_c} in the bilinear form of (2.7). We call T_{t_c,s_c} the *source-target translation operator*. In the extreme case $p = 0$, $T_{t_c,s_c} = 1/\|t_c - s_c\|_2$, and A_r is a rank-1 matrix. The FMM approximation in the extreme case is a simple interpolation scheme. In FMM terms, the product of V_s and a charge vector q is the vector of multipole expansion coefficients (with respect to the target points). The product of matrix T_{t_c,s_c} and vector $V_s q$ is the multipole-to-local translation (with respect to the target points). The last matrix-vector product with V_t^{T} gives the potential vector at the target points due to the charges at the source points.

The elementwise expansion is neither unique nor restricted to any specific coordinate system. It is more efficient to choose an expansion basis that makes use of harmonicity of the response function. In the proof of Theorem 2.1, we have chosen the multipole expansion used in the original version of the FMM by Greengard and Rokhlin in [13]. Anderson, among others, provided an alternative expansion method in [1], and Greengard and Rokhlin introduced yet another expansion method in the new version of the FMM [15]. The expansion-to-factorization approach illustrated in the proof of Theorem 2.1 can be applied to each of the expansions and results in a matrix factorization of the same form with the factors defined differently.

We have also exposed, in Theorem 2.1 and its proof, the symmetry between the source and the target in the elementwise expansion and hence the matrix factorization. Furthermore, the translation matrix T_{t_c,s_c} is symmetrical. The symmetrical structures originate from the symmetry between t and s in the response function $1/\|t-s\|$, although matrix A and its approximation A_r are not necessarily symmetric

in numerical values. In theory, a diagonal factorization of T_{t_c,s_c} leads to an elementwise expansion with diagonal source-target translation. In computational practice, the arithmetic complexity of the diagonalization method must be taken into account.

Before proceeding to discuss the FMM in the general case, we would like to comment on the truncation error introduced by the FMM approximation. In the FFT, there is no truncation error introduced in factoring the elements of a DFT matrix. In other words, the FFT factorizations are exact for the DFT. The FMM factorizations are approximate. Nonetheless, the FMM approximation can be made arbitrarily accurate in elementwise relative errors.

3. Levelwise Block Factorization.

3.1. Partition and Split. The matrix factorization of Theorem 2.1 requires that sources and targets be well separated in the sense of (2.1). When sources and targets are not well separated, we *divide* the sources (and the targets) into nonoverlapping subsets bounded by Cartesian boxes. The box size is $\delta > 0$ along every side. For any source point s, there is a source box centered at s_J so that $\|s - s_J\|_\infty \leq \delta/2$. A tie in the candidate boxes can be broken by a prespecified ordering in the box centers. We *partition* matrix A into blocks according to the box partition; namely, block A_{IJ} is the interaction matrix between the target box centered at t_I and the source box centered at s_J. We then *split* the block matrix into two components,

$$A = M + N, \tag{3.1}$$

where M consists of the matrix blocks with $\|t_I - s_J\|_\infty \geq \delta/\alpha$ for a fixed $\alpha < 1$ and N has the rest of the blocks. We call M the far-neighbor interaction matrix and N the near-neighbor interaction matrix.

According to Theorem 2.1, every nonzero block of matrix M can be approximated by a block matrix in factored form. By factoring out the common terms, we obtain an approximation to M in the block factorization form of (2.2).

THEOREM 3.1 (levelwise block factorization). *For any integer $p \geq 0$, the far-neighbor interaction matrix M is approximated by a block matrix of the factored form*

$$M_r = \operatorname{diag}(V_{t,I})^{\mathrm{T}} \, [T_{t_I,s_J}] \operatorname{diag}(V_{s,J}), \tag{3.2}$$

with the relative errors bounded elementwise as in (2.3). *Block $V_{\cdot,I}$ is the aggregation matrix of the pth-order local expansion vectors of the points in the Ith box. Block T_{t_I,s_J} is the source-target translation matrix between the source box centered at s_J and the target box centered at t_I.*

Assume that sources and targets are distributed randomly with uniform distribution in the initial bounded box. The near-neighbor matrix N has more zeros as the box size δ becomes smaller. Choose δ small enough so that N is sparse, i.e., so that the number of nonzero elements of N is $O(m + n)$, and big enough so that the number of source/target boxes is $O(m + n)$. Then, the direct evaluation of the matrix-vector product with N is of complexity $O(m + n)$. The matrix-vector product with M_r is carried out by matrix-vector products with $\operatorname{diag}(V_{s,J})$, $[T_{t_I,s_J}]$, and $\operatorname{diag}(V_{t,I})^{\mathrm{T}}$ consecutively. The complexity of the matrix-vector products with the local expansion vector matrices is $2r\eta(m + n)$ with $r = (p + 1)(2p + 1)$, where η is a constant determined by the algorithm to compute the local expansion vectors. Thus, the algorithm complexity of the computation with M_r is essentially determined by that of the computation with $[T_{t_I,s_J}]$. For simplicity we assume that $n \geq m$.

Let the box centers be chosen on a tensor grid. Since the numerical values of translation block T_{t_I,s_J} of order r depend only on the distance $t_I - s_J$, the block translation matrix $[T_{t_I,s_J}]$ can be embedded in a block Toeplitz matrix (in three dimensions) of order $O(n)$.

COROLLARY 3.2. *The arithmetic complexity of a matrix-vector product with M_r in the factored form* (3.2) *is $O(n \log n)$ via the use of the FFT.*

3.2. Nested Partition and Split. With the partition-split scheme described above, the error bound on the approximation errors in M_r is determined by the largest separation ratio α among the nonzero blocks. By Theorem 2.1 the separation ratio in the error bound per block decreases as the distance $\|t_I - s_J\|$ increases. The FMM exploits this decaying feature of the approximation errors by employing a hierarchical partition-split scheme and renders an efficient algorithm without using FFTs.

First, we choose the box size δ as big as possible in order to increase the block size in the far-neighbor interaction matrix M while the expansion order p is fixed and the separation ratio α is uniformly bounded. For instance, let $\alpha \le \sqrt{d}/(k+1)$ for some positive integer k and dimension d. Determine δ so that all the source and target points are enclosed in the initial box of size $(k+3)\delta$. Matrix A is partitioned into a block matrix of order at most $(k+3)^d$ in blocks. Split A into two terms,

$$A = \overline{M}_0 + N_0,$$

where $\overline{M}_0$ consists of the blocks satisfying the separation condition (2.1). Then, the source box and the target box of a nonzero block of $\overline{M}_0$ are at least k boxes apart. This is the partition-split at level 0.

Let M_0 be the approximate matrix to $\overline{M}_0$ in factored form (3.2). Consider the arithmetic complexity of the computation with M_0. The operation count of a matrix-vector product with $\mathrm{diag}(V_{t,I})$ or $\mathrm{diag}(V_{s,J})$ is $2rn$. The block translation matrix T_0 is of order $(k+3)^d$ in blocks, and each block is of order r. The operation count of a matrix-vector product with T_0 is therefore at most $2r^2(k+3)^{2d}$. The formation of the factors requires about the same number of operations.

If the near-neighbor matrix N_0 is not sparse, partition its blocks into smaller blocks. There are at most $(2k+1)^d$ nonzero blocks in each block row/column of N_0 at level 0. Let $\delta_1 = \delta/2$ and divide each source/target box into 2^d subboxes of size δ_1. Each block of N_0 is partitioned into $2^d \times 2^d$ subblocks. Split N_0 as $N_0 = \overline{M}_1 + N_1$. The nonzero blocks of $\overline{M}_1$ satisfy the condition (3.2) with the separation ratio bounded between $1/(2k+1)$ and $\sqrt{d}/(k+1)$. This is the partition-split at level $l = 1$.

Let M_1 be the approximation in the factored form to $\overline{M}_1$. Consider the computation with M_1. Again, the operation count of a matrix-vector product with each of the local expansion matrices is $2rn$. The block translation matrix T_1 is of order $2^{ld}(k+3)^d$ in blocks. In fact, T_1 has at most $2^d(2k+1)^d - (2k+1)^d = (2^d-1)(2k+1)^d$ nonzero blocks in each block row/column, *independent of the level number l*. The total operation count of a matrix-vector product with M_1 is no greater than $4rn + 2r^2 2^{ld}(2^d-1)(2k+1)^d(k+3)^d$. The formation of the factors requires about the same number of operations.

Repeat the partition-split process on N_l, $l = 1, 2, \ldots$, with $\delta_{l+1} = \delta_l/2$, up to level λ so that the computation with T_λ takes about the same number of operations as that with the local expansion matrices. For instance, if

$$2n \ge r 2^{d\lambda}(2^d-1)(2k+1)^d(k+3)^d > n 2^{1-d}, \tag{3.3}$$

then $\lambda < \log_{2^d} n$ and N_λ is sparse.

We summarize the above discussions in the following.

THEOREM 3.3 (nested partition and split). *Let k be a positive integer and $d = 3$. Assume that $(k+3)\delta$ is the size of the smallest box containing all the source and target points considered in matrix A of* (1.1). *Then,*

$$A = \sum_{l=0}^{\lambda} M_l + N_\lambda + E, \quad E = \sum_{l=0}^{\lambda} E_l, \quad |E(i,j)| \le A(i,j)\frac{1+\alpha}{1-\alpha}\alpha^{p+1},$$

where integer $p > 0$ is the expansion order and integer λ satisfies the conditions of (3.3). *For each $l = 1 : \lambda$, the far-neighbor matrix M_l at level l is of the factored form* (3.2), *with spatial partition size $\delta_l = \delta/2^l$ and the blockwise separation ratio α bounded between $1/(2k+1)$ and $\sqrt{d}/(k+1)$. The near-neighbor matrix N_λ at the finest partition level is sparse with $O(n)$ nonzero elements. The arithmetic complexity of a matrix-vector product with matrix $\sum_{l=0}^{\lambda} M_l + N_\lambda$ is $O(n \log n)$.*

The method with multilevel partition-split can be easily extended to the adaptive version [4], permitting various geometries in source and target domains. The method described in this section is an extension of the Barnes–Hut method [2], which uses a low-order multipole expansion at each partition level. The method here is symmetric; it recovers each of the dual versions (one is based on the multipole expansion at target points, the other at source points) by combining the translation matrix with the local expansion matrix on the source side or the target side. The symmetric version has a few advantages: (i) In addition to the concurrency in computations among all λ levels, the local expansion vector matrices and the translation matrix at every level can be formed simultaneously. (ii) When targets and sources coincide, the local expansion matrices are the same and the translation matrix is symmetric at every level. (iii) Updates in the source point distribution and/or in the target point distribution can be easily handled.

A computational issue with translation matrix T_l is the specification of the nonzero blocks in each block row/column at every partition level l. The block row of T_l corresponding to a target box has nonzero blocks in the block columns corresponding to the source boxes that are at least k boxes away at level l but are within a k-box neighborhood at level $l-1$. Every box i at level $l-1$ is divided into 2^d child boxes at level l. We use 2^d stencils to specify the block structure of T_l, $l > 0$, one for each child box. The stencils vary with k. We illustrate in Figure 3.1 the one-dimensional stencils (the even stencil and the odd stencil) for the nonzero blocks of T_l in each block row/column with $k = 1, 2$. We illustrate a two-dimensional stencil for each of the cases $k = 1, 2$. Every d-dimensional stencil is formed from d one-dimensional stencils by the tensor product with logical operation OR. In the adaptive FMM [4], the stencil for every target box is masked with the distribution of nonempty source boxes in the neighborhood. In FMM terms, the masked stencil is the interaction list.

4. Nested Factorization and Multiplication.

In this section we introduce the FMM algorithm of complexity $O(n)$, which is asymptotically optimal, up to a constant.

4.1. Nested Translation.

The factorization of the far-neighbor matrix M_l introduced in the last section requires the local expansions of every source/target point with respect to its box center at level l. The following theorem relates the local expansion matrices at level l, $l < \lambda$, to those at the finest partition level λ.

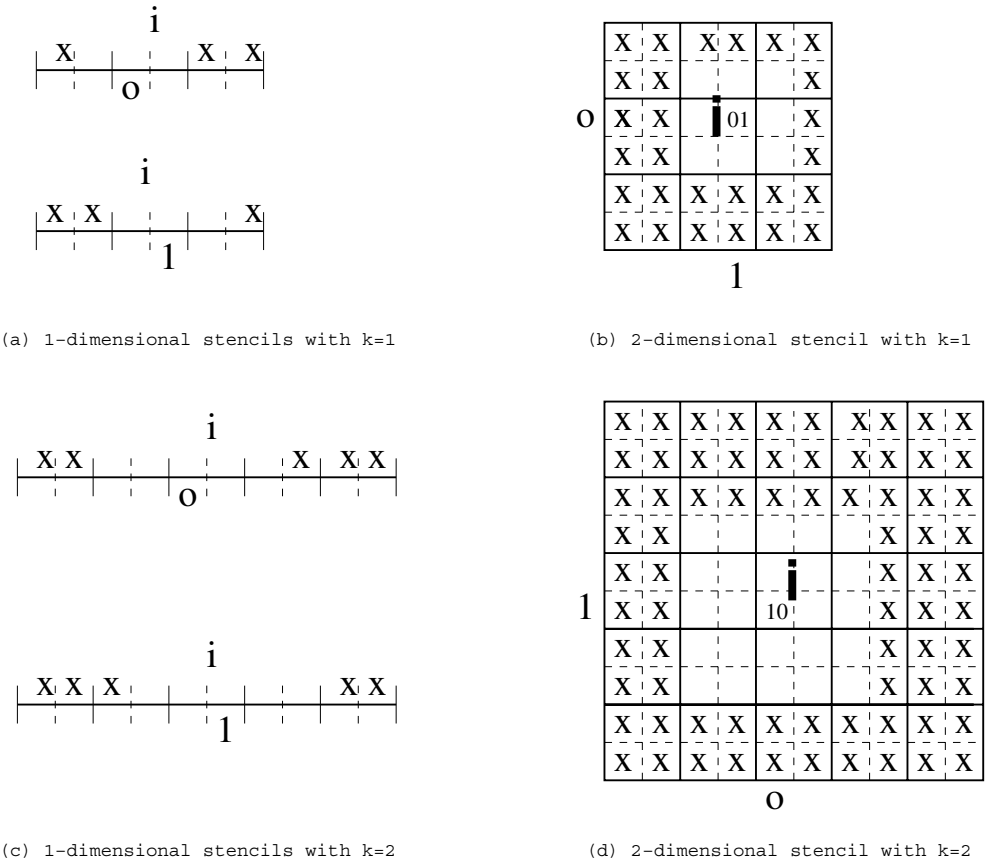

Fig. 3.1 *The basic stencils and stencil construction.*

THEOREM 4.1 (nested translation). *Let* $M_l = \mathrm{diag}(V_{t,I}|l)^{\mathrm{T}} T_l \,\mathrm{diag}(V_{s,J}|l)$ *be the far-neighbor matrix at level* l *defined as in Theorem* 3.3. *Then,*

$$M_l = \mathrm{diag}(V_{t,I}|\lambda)^{\mathrm{T}} (P_{\lambda-1}^{\mathrm{T}} \cdots P_l^{\mathrm{T}})\, T_l\, (P_l \cdots P_{\lambda-1}) \mathrm{diag}(V_{s,J}|\lambda),$$

where

$$P_j \stackrel{\mathrm{def}}{=} I_j \otimes [C_{1,j} C_{2,j}, \ldots, C_{8,j}], \quad j = l : \lambda - 1,$$

translates and aggregates the local expansion vectors from level $j+1$ *to level* j, I_j *is the identity matrix of order equal to the number of boxes at level* j, *and* $C_{i,j}$ *is the translation matrix from the* i*th child box to the parent box at level* j.

Proof. Let s_c be the center of source s_i at level $j+1$ and $\bar{s}_c$ be the center of s_i at level j. We have $s_i - \bar{s}_c = (s_i - s_c) - (\bar{s}_c - s_c)$. By the third addition theorem of spherical harmonic functions [9],

$$\begin{aligned} Z_{n,m}(s_i - \bar{s}_c) &= \sum_{l=0}^{n} \sum_{k=-l}^{l} (-1)^l Z_{l,k}(\bar{s}_c - s_c) Z_{n-l,m-k}(s_i - s_c) \\ &= \sum_{l=0}^{p} \sum_{k=-p}^{p} (-1)^{n-l} Z_{n-l,m-k}(\bar{s}_c - s_c) Z_{l,k}(s_i - s_c), \end{aligned}$$

where $Z_{n,m}$ is defined as in the proof of Theorem 2.1. With a similar approach, we obtain the following relationship between the local expansion vectors of s_i at the two levels,

$$v(s_i - \bar{s}_c) = C\, v(s_i - s_c),$$

where the translation matrix C is defined as follows:

$$C(i,j) = (-1)^{n-l} Z_{n-l,m-k}(\bar{s}_c - s_c), \qquad \begin{array}{l} i = n(2p+1) + (p+1) + m, \\ j = l(2p+1) + (p+1) + k. \end{array}$$

The matrix $[C_{1,j}, \ldots, C_{8,j}]$ aggregates the translations from eight children to the parent. The matrix P_j represents the uniform operation of all parent-children translations from level $j+1$ to level j. We start with the local expansion at the finest partition level λ and apply the translation successively to level l. The target local expansion vectors at different levels share the same translation relationship. ∎

4.2. Nested Multiplication. We now consider the computation of a matrix-vector product with matrix $\sum_j M_j$, exploiting the nested factorization structure. Let q be the charge vector at the source points. Define

$$\begin{aligned} q_\lambda &= \operatorname{diag}(V_{s,J}|\lambda)\, q, \\ q_l &= P_l\, q_{l+1}, \qquad l = \lambda - 1 : -1 : 0. \end{aligned} \tag{4.1}$$

Then, by Theorem 4.1,

$$\sum_{l=0}^{\lambda} M_l\, q = \operatorname{diag}(V_{t,I}|\lambda)^{\mathrm{T}} \sum_{l=0}^{\lambda} P_{\lambda-1}^{\mathrm{T}} \cdots P_l^{\mathrm{T}} T_l\, q_l.$$

Applying Horner's rule, we have

$$\begin{aligned} p_0 &= T_0\, p_0, \\ p_l &= P_l^{\mathrm{T}} p_{l-1} + T_l\, q_l, \qquad l = 1 : \lambda, \\ \textstyle\sum_{l=0}^{\lambda} M_l\, q &= \operatorname{diag}(V_{t,I}|\lambda)\, p_\lambda. \end{aligned} \tag{4.2}$$

The computation of (4.1) is a bottom-up sweep, and the computation of (4.2), a top-down sweep. In FMM terms, (4.1) are the multipole-to-multipole translations, $T_l q_l$ are the multipole-to-local translations, and $P_l^{\mathrm{T}} p_{l-1}$ are the local-to-local translations.

COROLLARY 4.2 (nested multiplication). *The operation count of a matrix-vector product with* $\sum_{l=0}^{\lambda} M_l$ *by* (4.1) *and* (4.2) *is* $O(n)$.

Proof. In (4.1), the computation of q_λ takes $O(rn)$ operations. Translating q_{l+1} to q_l takes $8^{l+1} 2r^2 (k+2)^d$ operations, $l = \lambda{-}1 : -1 : 0$. In total, the computation of (4.1) needs $O(n)$ operations since $\lambda < \log_8(n)$. A similar argument applies to the computation of (4.2). ∎

The nested version reduces the arithmetic complexity to $O(n)$ and introduces a sequential data dependency from level to level. The introduced data dependency, however, does not preclude the use of pipelining techniques. Moreover, every local translation has the parallelism described by the Kronecker product.

5. Conclusion. We have discussed a number of methodologies in designing algorithms, based on FMM ideas, for fast computation of a dense matrix-vector product with matrix (1.1): (a) From an elementwise expansion in the bilinear form of (2.7) one can derive a symmetrical factorization of an interaction matrix (block) that satisfies the separation condition of Theorem 2.1. (b) When the response function is translation invariant, a matrix partition-split at a fine partition level leads to an efficient FFT-based algorithm of complexity $O(n \log n)$. (c) When the approximation errors decay with distance, the partition-split scheme may be used at multiple levels and results in an efficient levelwise algorithm of complexity $O(n \log n)$. This version enables the use of an adaptive scheme. (d) By introducing a translation relationship between the local expansion vectors at different levels, one obtains an even more efficient algorithm of complexity $O(n)$.

We have also revealed the matrix factor relations among a few basic algorithm variants of the FMM. In particular, we have underlined that the translation blocks can be diagonalized either numerically or analytically. Diagonalizing the $r \times r$ blocks reduces the constant r^2 in the arithmetic complexity. In [8] Elliott and Board diagonalized translation blocks via the use of FFTs. The FFT-based diagonalization approach becomes more sensitive to roundoff errors as the size of the translation blocks increases with the expansion order p. An analytical diagonalization approach given recently by Greengard and Rokhlin in [15] is not sensitive to the roundoff errors.

Kapur and Zhao [16] use a numerical approach to obtaining a blockwise factorization when the response function in question is not analytically available. The block partition and ordering schemes we have described in this paper actually clarify the partition and ordering scheme considered heuristic in [16] as well.

It is our hope that the matrix version will play a similar role in FMM implementations as the FFT matrix theory [22] does in FFT implementations. Similar to the FFT, there are many FMM factorizations to choose from, and each factorization changes with the partition scheme and expansion scheme. FMM implementations vary more with accuracy requirements, source/target distributions, and geometries.

For the original FMM ideas, we refer the reader to the well-recognized FMM paper [13] by Greengard and Rokhlin, which was reprinted in 1997 [14], Greengard's dissertation [9, 10], and the paper [4] by Carrier, Greengard, and Rokhlin on the adaptive FMM. Recent publications by Greengard and Rokhlin [15] and by Cheng, Greengard, and Rokhlin [6] described the most efficient implementations to date with respect to arithmetic complexity. For references to other works employing FMM ideas, we refer the reader to the bibliography database at http://www.netlib.org/bibnet/authors/f/fastmultipole.bib.

Acknowledgment. The authors thank the Defense Sciences Office of DARPA for its support.

REFERENCES

[1] C. R. Anderson, *An implementation of the fast multipole method without multipoles*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 923–947.

[2] J. E. Barnes and P. Hut, *A hierarchical $O(N \log N)$ force-calculation algorithm*, Nature, 324 (1986), pp. 446–449.

[3] J. A. Board, Jr., *Introduction to a fast algorithm for particle simulations*, J. Comput. Phys., 135 (1997), p. 279.

[4] J. Carrier, L. Greengard, and V. Rokhlin, *A fast adaptive multipole algorithm for particle simulations*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 669–686.

[5] H. Cheng and L. Greengard, *A method of images for the evaluation of electrostatic fields in systems of closely spaced conducting cylinders*, SIAM J. Appl. Math., 58 (1998), pp. 122–141.

[6] H. Cheng, L. Greengard, and V. Rokhlin, *A fast adaptive multipole algorithm in three dimensions*, J. Comput. Phys., 155 (1999), pp. 468–498.

[7] W. C. Chew, J. M. Jin, C. C. Lu, E. Michielssen, and J. M. Song, *Fast solution methods in electromagnetics*, IEEE Trans. Antennas Propagation, 45 (1997), pp. 533–543.

[8] W. D. Elliott and J. A. Board, Jr., *Fast Fourier transform accelerated fast multipole algorithm*, SIAM J. Sci. Comput., 17 (1996), pp. 398–415.

[9] L. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems*, Ph.D. thesis, Yale University, New Haven, CT, 1987.

[10] L. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems*, ACM Distinguished Dissertations, MIT Press, Cambridge, MA, 1988.

[11] L. Greengard, *Fast algorithms for composite materials*, in Materials Research Society Symposium Proceedings, Materials Theory, Simulations and Parallel Algorithms 408, Materials Research Society, Warrendale, PA, 1996, pp. 93–97.

[12] L. Greengard and J. Helsing, *On the numerical evaluation of elastostatic fields in locally isotropic two-dimensional composites*, J. Mech. Phys. Solids, 46 (1998), pp. 1441–1462.

[13] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys., 73 (1987), pp. 325–348.

[14] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys., 135 (1997), pp. 280–292.

[15] L. Greengard and V. Rokhlin, *A new version of the fast multipole method for the Laplace equation in three dimensions*, Acta Numer., 6 (1997), pp. 229–269.

[16] S. Kapur and J. Zhao, *A fast method of moments solver for efficient parameter extraction of MCMs*, in 34th Design Automation Conference, MP Associates, Boulder, CO, 1997, pp. 141–146.

[17] S. Krishnan and L. V. Kale, *A parallel adaptive fast multipole algorithm for n-body problems*, in Proceedings of the International Conference on Parallel Processing, 1995, pp. 46–50.

[18] V. Rokhlin, *Rapid solution of integral equations of classical potential theory*, J. Comput. Phys., 60 (1985), pp. 187–207.

[19] J. Salmon and M. S. Warren, *Parallel, out-of-core methods for N-body simulation*, in Proceedings of the Eighth SIAM Conf. on Parallel Processing for Scientific Computing, SIAM, Philadelphia, 1997.

[20] B. Shanker, S.-K. Han, E. Michielssen, and W. C. Chew, *A fast multipole approach to computing scattering from an inhomogeneous bianisotropic cylindrical object using Beltrami fields*, in IEEE Antennas and Propagation Society International Symposium, 1997, pp. 43–51.

[21] R. Strebel, *Pieces of Software for the Coulombic m Body Problem*, Ph.D. thesis, ETH Zurich, Switzerland, 2000.

[22] C. Van Loan, *Computational Frameworks for the Fast Fourier Transforms*, Frontiers Appl. Math. 10, SIAM, Philadelphia, 1992.

[23] F. Zhao and S. L. Johnsson, *The parallel multipole method on the connection machine*, SIAM J. Sci. Statist. Comput., 12 (1991), pp. 1420–1437.