COMMUNICATIONS IN COMPUTATIONAL PHYSICS
Vol. **5**, No. 1, pp. 108-125

Commun. Comput. Phys.
January 2009

Comparison and Improvement of the Computational Efficiencies of Two FFT-Based Iterative Solution Methods for the Scalar Multiple-Scattering Equation

Kristopher T. Kim*

Electromagnetic Scattering Branch, Sensors Directorate, Air Force Research Laboratory, Hanscom AFB, MA 01731-3010, USA.

Received 4 September 2007; Accepted (in revised version) 30 April 2008

Communicated by Wei Cai

Available online 15 July 2008

Abstract. We consider two existing FFT-based fast-convolution iterative solution techniques for the scalar T-matrix multiple-scattering equation [1]. The use of the FFT operation requires field values be expressed on a regular Cartesian grid and the two techniques differ in how to go about achieving this. The first technique [6,7] uses the non-diagonal translation operator [1,9] of the spherical multipole field, while the second method [11] uses the diagonal translation operator of Rokhlin [10]. Because of its use of the non-diagonal translator, the first technique has been thought to require a greater number of spatial convolutions than the second technique. We establish that the first method requires only half as many convolution operations as the second method for a comparable numerical accuracy and demonstrate, based on an actual CPU time comparison, that it can therefore perform iterations faster than the second method. We then consider the respective symmetry relations of the non-diagonal and diagonal translators and discuss a memory-reduction procedure for both FFT-based methods. In this procedure, we need to store only the minimum sets of near-field and far-field translation operators and generate missing elements on the fly using the symmetry relations. We show that the relative cost of generating the missing elements becomes smaller as the number of scatterers increases.

AMS subject classifications: 74J20, 81010

Key words: Multiple scattering, iterative method, FFT, symmetry.

1 Introduction

The scattering of acoustic waves from an ensemble of discrete scatterers is an important, age-old problem that finds applications in areas as diverse as sonar and medical

*Corresponding author. *Email address:* trout@ieee.org (K. T. Kim) imaging. In many applications, multiple scattering effects cannot be ignored in determining the overall scattering characteristics, but their numerical determination can be computationally challenging. They can be determined, for example, by solving the governing multiple-scattering equation [1] based on the T-matrix formalism [2]. If there are N_p scatterers in the ensemble and each scatterer requires P partial waves to represent its scattering behavior, one then needs to solve a system of linear equations of size N by N, where $N \equiv PN_p$. When N is modest, one may be able to solve the matrix equation using either a direct method such as the LU decomposition or an iterative technique such as the conjugate gradient (CG) method [3,4]. A direct method would require $\mathcal{O}(N^3)$ floating point operations (FPOs) and $\mathcal{O}(N^2)$ memory units (MUs), while an iterative method would require $\mathcal{O}(N^2)$ FPOs per iteration and $\mathcal{O}(N^2)$ MUs. However, as N increases, with these unfavorable computational complexities, one quickly finds oneself unable to store the matrix even on a supercomputer, let alone solve the matrix equation.

Significant efforts have been directed at reducing the aforementioned undesirable computational complexities over the last decade and a half, and a number of efficient algorithms have been proposed. Among the useful algorithms that have emerged is the FFT T-matrix technique [6,7] that was originally developed for electromagnetic scattering applications. This technique was later adopted by [11] for the acoustic case as part of its comparative study on efficient prediction techniques for scattering from a cluster of particles. The idea is to use the FFT-based fast-convolution technique to expedite matrix-vector multiplications, which are the most costly part of an iterative solution process.

In this paper we compare and improve the computational efficiencies of two existing FFT T-matrix methods for the scalar T-matrix multiple-scattering equation [1]. The use of the FFT operation requires field values be expressed on a regular Cartesian grid and the two versions differ in how to go about achieving this. The first method [6,7,11] uses the non-diagonal translation operator [1,9] of the spherical multipole field, while the second method [11] uses the diagonal translation operator of Rokhlin [10]. Because of its use of the non-diagonal translator, the first version is thought to require a greater number of spatial convolutions than the second one [11]. We first establish that the first method requires half as many spatial convolutions as the second method for a given numerical accuracy and show, based on a CPU time comparison, that the first method indeed performs matrix-vector multiplications faster.

As alluded to earlier, when solving scattering from a volume distribution of scatterers, it is often the memory requirement that sets the upper limit on the problem size that can be solved on a given computer. Therefore, it is highly desirable to be able to reduce the memory requirement. Since storage of the translation matrix (TM) drives the overall memory requirements of the FFT T-matrix methods [13], we discuss the respective symmetry relations of the non-diagonal and diagonal TMs and develop a memory-reduction procedure for both FFT T-matrix methods. The procedure requires only the minimum sets of the translators be stored and generates missing elements from them on the fly using the symmetry relations. We show that the procedure reduces the memory requirements of both FFT T-matrix methods by non-trivial factors. These symmetry relations can easily be implemented in an existing FFT T-matrix code, allowing it to solve larger problems.

The accuracy of the FFT T-matrix method improves with the increasing number of partial waves, $P \equiv ({L_{ex}}^2+1)^2$, used in the expansion of the scattered field and the increasing number of multipole terms, $T \equiv ({L_T}^2+1)^2$, retained in the translation to the Cartesian grid [6], [7], [11]. As we will discuss, the overall memory requirement scales as $\mathcal{O}({L_{ex}}^4 N_p)+\mathcal{O}({L_T}^4 N_G)$ for the FFT method based on the non-diagonal translator and $\mathcal{O}({L_{ex}}^4 N_p)+\mathcal{O}({L_T}^2 N_G)$ for the FFT method based on the diagonal translator, where N_p is the number of scatterers in the ensemble and N_G is the size of the extended regular Cartesian grid. Therefore, accurate multiple-scattering predictions require substantially more memory and the memory-reduction procedure discussed in this work can alleviate the increased memory requirement.

The organization of the paper is as follows. In Section 2, we briefly review the scalar T-matrix multiple-scattering equation [1] to establish the conventions and notations used in the paper. In Section 3, we establish the CPU and memory requirements of the existing FFT T-matrix method based on the non-diagonal translator, and examine the respective symmetry relations the non-diagonal TM satisfies in configuration- and k-spaces. Specifically, we show that the CPU time requirement scales as $\mathcal{O}({L_T}^2 N_G \log_2 N_G)$ and that the symmetry relations reduce the near-field TM memory requirement by almost eightfold and the far-field TM requirement by nearly sixteenfold. In Section 4, after establishing the CPU and memory requirements of the existing FFT T-matrix method based on the diagonal translator [11], we discuss the symmetry relations the diagonal TM satisfies in k-space and show that they reduce the far-field TM memory requirement by eightfold. Since the near-field TM remains un-diagonalized, we may use the same configuration-space symmetry relations discussed in Section 3 and reduce the near-field TM memory requirement by nearly eightfold. In Section 5, we first validate the convolution- and memory-reduction techniques discussed in Sections 3 and 4 by computing the bistatic acoustic cross section of an ensemble of spheres and show that their use introduces no numerical inaccuracy. We then compare the CPU times required by the two FFT T-matrix methods to perform matrix-vector multiplications and show that the FFT method based on the non-diagonal translator performs matrix-vector multiplications faster. We also show that the relative cost of using the symmetry relations to generate missing translation matrix elements on the fly decreases as the number of unknowns increases. Section 6 concludes the paper. All the symmetry relations of the non-diagonal TM used in this paper are derived in [12] except (3.4), whose derivation is provided in the appendix.

2 T-matrix multiple scattering equation

We briefly review the T-matrix multiple-scattering equation [1,9] for the purpose of establishing the notations and conventions used in the paper and examine the computational cost of solving the equation as a function of the number of scatterers, N_p, in the ensemble and the order of partial waves, L_{ex}, used in the expansion of the incident and scattered fields. Let us consider the scattering of a plane wave, $\phi^{inc}(\bar{r}) = e^{i\bar{k}\cdot\bar{r}}$, from an ensemble of N_p acoustically small, non-overlapping, identical spherical scatterers with radius a located at $\bar{r}_j$, where $j=1,\cdots,N_p$. $\phi^{inc}(\bar{r})$ may be expanded in terms of "regular" multipole fields, $Rg\phi(\bar{r})$,

$$\phi^{inc}(\bar{r}) = \sum_{l=0}^{L_{ex}} \sum_{m=-l}^{l} a_{l,m} \cdot Rg\phi_{l,m}(\bar{r}).$$

Here, $Rg\phi_{l,m}(\bar{r}) \equiv j_l(kr)Y_{l,m}(\hat{r})$, where $j_l(kr)$ is the spherical Bessel function and $Y_{l,m}(\hat{r})$ the spherical harmonics with $\hat{r} \equiv (\theta,\phi)$ and k the wavenumber. $L_{ex} = \mathrm{Int}(ka+n)$, where n is a small integer and Int stands for the integer function. The scattered field at $\bar{r}$ from the jth scatterer, $\phi_j^{sc}(\bar{r})$, may be expanded in a similar way,

$$\phi_j^{sc}(\bar{r}) = \sum_{l=0}^{L_{ex}} \sum_{m=-l}^{l} b_{l,m}^{(j)} \cdot \phi_{l,m}(\bar{r}-\bar{r}_j).$$

Here,

$$\phi_{l,m}(\bar{r}-\bar{r}_j) \equiv \phi_{l,m}(\bar{\rho}_j) \equiv h_l(k\rho_j)Y_{l,m}(\hat{\rho}_j),$$

where $h_l(k\rho_j)$ is the spherical Hankel function of the first kind and $\bar{\rho}_j \equiv \bar{r}-\bar{r}_j$. The total scattered field, $\phi^{sc}(\bar{r})$, is then

$$\phi^{sc}(\bar{r}) = \sum_{j=1}^{N_p} \phi_j(\bar{r}-\bar{r}_j).$$

The unknown expansion coefficients, $b_{lm}^{(i)}$, are related to the incident wave expansion coefficients of the jth sphere, $a_{lm}^{(j)}$, and the isolated single-particle T matrix of the jth particle, $t_{lm}^{(j)}$, through

$$b_{l',m'}^{(j)} - t_{l',m'}^{(j)} \sum_{l=0}^{L_{ex}} \sum_{m=-l}^{l} \sum_{i\neq j}^{N_p} \alpha_{l',m'}^{l,m}(\bar{r}_j-\bar{r}_i) \cdot b_{l,m}^{(i)} = t_{l',m'}^{(j)} a_{l',m'}^{(j)}. \tag{2.1}$$

Here, $\alpha_{l',m'}^{l,m}(\bar{r}_j-\bar{r}_i)$ is the translation matrix of the spherical multipole field [1,9] and acts as a coupling coefficient (propagator) between two spatially separated multipole fields, $\phi_{l,m}(\bar{r}_i)$ and $\phi_{l',m'}(\bar{r}_j)$. Expressions for $t_{l',m'}^{(j)}$ for scattering from a penetrable sphere and an impedance sphere can be found, for example, in [5] and expressions for scattering from a rigid sphere can be deduced as a limiting case.

We may write (2.1) in matrix form

$$[\mathbf{I}-\mathbf{T}\cdot\mathbf{A}]\cdot\bar{b} = \mathbf{T}\cdot\bar{a}, \tag{2.2}$$

where $\mathbf{I}$, $\mathbf{T}$ and $\mathbf{A}$ are, respectively, the identity, transition and translation matrices. Since $[\mathbf{I}-\mathbf{T}\cdot\mathbf{A}]$ is of size N by N with $N \equiv PN_p \equiv (L_{ex}+1)^2 N_p$, a direct solution of (2.2) using, for example, the LU decomposition of $[\mathbf{I}-\mathbf{T}\cdot\mathbf{A}]$ would require $\mathcal{O}(N^3)$ FPOs and $\mathcal{O}(N^2)$ MUs

[3]. An iterative solution based on, for example, the CG method [4] would require $\mathcal{O}(N^2)$ FPOs per iteration and $\mathcal{O}(N^2)$ MUs. These unfavorable CPU and memory requirements severely limit the size of the problems that can be solved by the conventional methods.

3 FFT T-matrix method with non-diagonal translator

3.1 Original formulation

In order to reduce the aforementioned computational complexities, [6] and [7] were the first to apply the FFT-based fast-convolution technique to the T-matrix multiple-scattering equation. Their technique exploits the translation property of the translation coefficients of the electromagnetic multipole fields to express field values on a regular Cartesian grid. It was adopted by [11] for the acoustic case as part of its comparative study on efficient prediction techniques for acoustic scattering from a cluster of particles. Koc and Chew [11] showed that the FFT-based fast-convolution technique reduces the cost of performing the spatial convolution in (2.1) from $\mathcal{O}(N_p^2)$ to $\mathcal{O}(N_G \log_2 N_G)$ FPOs for each (l,m) and (l',m') combination and the overall memory requirement from $\mathcal{O}(L_{ex}{}^4 N_p{}^2)$ to $\mathcal{O}(L_{ex}{}^4 N_p)+\mathcal{O}(L_T{}^4 N_G)$, where N_G is the size of the extended regular Cartesian grid containing the scatterers and $(L_T+1)^4$ is the total number of modal combinations of the translation coefficient. (See (3.1).) Thus, whenever the particle distribution supports a regular grid with $\mathcal{O}(N_G \log_2 N_G)<\mathcal{O}(N_p{}^2)$, the FFT-based fast-convolution technique can perform the spatial convolution in (2.1) faster using less memory. A moderately uniform, dense particle distribution supports a regular grid that satisfies $N_G \log_2 N_G \ll N_p^2$. As the particle distribution becomes more inhomogeneous or tenuous, the FFT-based convolution technique becomes increasingly inefficient since a highly inhomogeneous or tenuous particle distribution may not support a grid with $\mathcal{O}(N_G \log_2 N_G) < \mathcal{O}(N_p^2)$. For this type of particle distribution, an adaptive method such as the one studied in [11] is more efficient. It is, however, important to note that, while the FFT-based method and the adaptive method are capable of performing the spatial convolution with $\mathcal{O}(N_G \log_2 N_G)$ and $\mathcal{O}(N_p \log_2 N_p)$ FPOs, respectively, the proportionality constant for the former is substantially smaller than that of the latter. In addition, efficient FFT routines [8] exist and some are highly optimized on many computing platforms to take advantage of their special hardware or software features and thus FFT-based methods are able to perform convolution operations quite efficiently. However, for a highly inhomogeneous or tenuous particle distribution[†], an adaptive technique must be used.

The FFT-based fast-convolution method requires field values be specified on a regular Cartesian grid. Thus, when the particles are not evenly spaced, it is necessary to express $\alpha_{l',m'}^{l,m}(\bar{r}_j-\bar{r}_i)$ in terms of $\alpha_{l',m'}^{l,m}(\bar{R}_j-\bar{R}_i)$, where $\bar{R}_j$ and $\bar{R}_i$, as shown in Fig. 1, are the grid

[†]For truly tenuous distributions, multiple scattering effects are small and thus may be ignored.

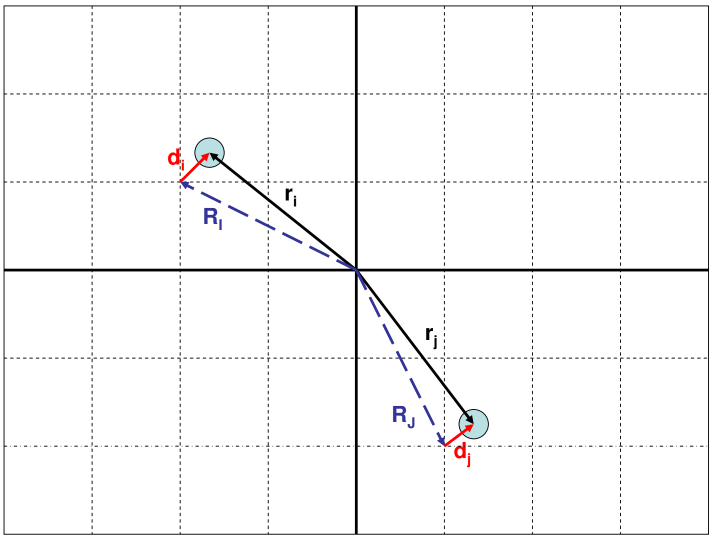

Figure 1: A regular Cartesian grid showing the translation to the nearest grid points.

points nearest to $\bar{r}_j$ and $\bar{r}_i$, respectively. We let

$$\bar{r}_i \equiv \bar{R}_i + \bar{d}_i, \quad \bar{r}_j \equiv \bar{R}_j + \bar{d}_j,$$

where $|\bar{R}_j| > |\bar{d}_j|$ and $|\bar{R}_i| > |\bar{d}_i|$. Using the translation property of $\alpha^{l,m}_{l',m'}(\bar{r}_j - \bar{r}_i)$ [1],

$$\alpha^{l,m}_{l',m'}(\bar{r}_j - \bar{r}_i) = \sum_{L=0}^{L_T} \sum_{M=-L}^{L} \beta^{l,m}_{L,M}(\bar{d}_j) \sum_{L'=0}^{L_T} \sum_{M'=-L'}^{L'} \alpha^{L,M}_{L',M'}(\bar{R}_j - \bar{R}_i) \beta^{L',M'}_{l',m'}(-\bar{d}_i), \tag{3.1}$$

where $\beta^{l,m}_{L,M}(\bar{d}_j)$ and $\beta^{L',M'}_{l',m'}(-\bar{d}_i)$ are the same as $\alpha^{l,m}_{L,M}(\bar{d}_j)$ and $\alpha^{L',M'}_{l',m'}(-\bar{d}_i)$ except that they are regular at the origin [1,9], and $L_T = \text{Int}(kd+n)$ with $d = \max\ (d_i, d_j)$ and n being a small integer.

The spatial convolution in (2.1) may be separated into two parts depending on the distance, $r_{ji} \equiv |\bar{r}_j - \bar{r}_i|$:

$$\begin{aligned}
\Gamma_{l',m'}(\bar{r}_j) &\equiv \sum_{l,m}^{L_{ex}} \sum_{i \neq j} \alpha^{l,m}_{l',m'}(\bar{r}_j - \bar{r}_i) \cdot b^{(i)}_{l,m} \\
&= \sum_{l,m}^{L_{ex}} \left[\sum_{i \in \mathcal{N}_i} \alpha^{l,m}_{l',m'}(\bar{r}_j - \bar{r}_i) \cdot b^{(i)}_{l,m} + \sum_{i \in \mathcal{F}_i} \alpha^{l,m}_{l',m'}(\bar{r}_j - \bar{r}_i) \cdot b^{(i)}_{l,m} \right] \\
&\equiv \Gamma^{(N)}_{l',m'}(\bar{r}_j) + \Gamma^{(F)}_{l',m'}(\bar{r}_j),
\end{aligned} \tag{3.2}$$

where $\mathcal{N}_i$ and $\mathcal{F}_i$, respectively, represent the particles in the near field ($r_{ji} \leq \delta$) and in the far field ($r_{ji} > \delta$) of the ith particle for some suitably chosen δ.

The near-field convolution, $\Gamma^{(N)}_{l',m'}(\bar{r}_j)$, requires $\mathcal{O}(L_{ex}{}^4 N_p)$ FPOs for all j and (l',m') using precomputed $\alpha^{l,m}_{l',m'}(\bar{r}_j-\bar{r}_i)$ with $r_{ij}<\delta$, for each particle has an $\mathcal{O}(1)$ number of particles in its near field and there are $(L_{ex}+1)^4$ combinations of (l,m) and (l',m'). Similarly, it takes $\mathcal{O}(L_{ex}{}^4 N_p)$ MUs to store the precomputed near-field translation coefficients.

The far-field convolution, $\Gamma^{(F)}_{l',m'}(\bar{r}_j)$, can be expedited using (3.1) and the FFT-based convolution technique:

$$\begin{aligned}
\Gamma^{(F)}_{l',m'}(\bar{r}_j) &= \sum_{L',M'}^{L_T} \beta^{L',M'}_{l',m'}(\bar{d}_j) \sum_{L,M}^{L_T} \sum_i \hat{\alpha}^{L,M}_{L',M'}(\bar{R}_j-\bar{R}_i) \sum_{l,m}^{L_{ex}} \beta^{l,m}_{L,M}(-\bar{d}_i)\cdot b^{(i)}_{l,m} \\
&= \sum_{L',M'}^{L_T} \beta^{L',M'}_{l',m'}(\bar{d}_j) \sum_{L,M}^{L_T} FFT^{-1}\left[\tilde{\alpha}^{L,M}_{L',M'}(\bar{q})\tilde{\Psi}_{L,M}(\bar{q})\right] \\
&= \sum_{L',M'}^{L_T} \beta^{L',M'}_{l',m'}(\bar{d}_j) FFT^{-1}\left[\sum_{L,M}^{L_T} \tilde{\alpha}^{L,M}_{L',M'}(\bar{q})\tilde{\Psi}_{L,M}(\bar{q})\right],
\end{aligned} \tag{3.3}$$

where $\hat{\alpha}^{L,M}_{L',M'}(\bar{R}_j-\bar{R}_i)$ is defined according to

$$\hat{\alpha}^{L,M}_{L',M'}(\bar{R}_j-\bar{R}_i) \equiv \begin{cases} \alpha^{L,M}_{L',M'}(\bar{R}_j-\bar{R}_i) & \text{if } |\bar{R}_j-\bar{R}_i|>\delta, \\ 0 & \text{otherwise.} \end{cases}$$

In (3.3), FFT^{-1} represents the inverse fast Fourier transform (IFFT); $\tilde{\alpha}^{L,M}_{L',M'}(\bar{q})$ and $\tilde{\Psi}_{L,M}(\bar{q})$ are, respectively, the Fourier transforms of $\hat{\alpha}^{L,M}_{L',M'}(\bar{R}_j-\bar{R}_i)$ and $\Psi_{L,M}(-\bar{d}_i)$ where

$$\Psi_{L,M}(-\bar{d}_i) \equiv \sum_{l,m} \beta^{l,m}_{L,M}(-\bar{d}_i)\cdot b^{(i)}_{l,m};$$

and $\bar{q}$ is the Fourier conjugate variable in k-space. $\tilde{\alpha}^{L,M}_{L',M'}(\bar{q})$ are pre-computed for all combinations of (L,M) and (L',M') at the computational cost of $\mathcal{O}(L_T{}^4 N_G \log_2 N_G)$ FPOs and $\mathcal{O}(L_T{}^4 N_G)$ MUs. Each iteration updates the unknown expansion coefficients $b^{(i)}_{l,m}$ and thus $\tilde{\Psi}_{L,M}(\bar{q})$ also needs to be updated at the cost of $\mathcal{O}(L_T{}^2 N_G \log_2 N_G)$. The convolution, $\Gamma^{(F)}_{l',m'}(\bar{r}_j)$, then requires $\mathcal{O}(L_T{}^2 N_G \log_2 N_G)$ FPOs with the precomputed $\tilde{\alpha}^{L,M}_{L',M'}(\bar{q})$ when N_p is large.

3.2 Memory requirement reduction

With a three-dimensional volume distribution of particles, it is often the memory requirement, rather than the CPU requirement, that determines the size of the largest problem that can be solved on a given computer. Therefore, it is important to be able to reduce the memory requirement. As mentioned previously, the storage of the far-field and near-field TMs, which respectively require $\mathcal{O}(L_T{}^4 N_G)$ and $\mathcal{O}(L_{ex}^4 N_p)$ MUs, drives the overall memory requirement of the FFT-based fast-convolution method. In this section, we show that it is possible to further reduce both the near- and far-field TM storage requirements by exploiting the respective symmetry relations [12] of $\alpha^{l,m}_{l',m'}(\bar{r}_j-\bar{r}_i)$ and $\tilde{\alpha}^{l,m}_{l',m'}(\bar{q})$. Our approach is similar to the one adopted in [13] for the electromagnetic multiple-scattering equation.

As discussed in [13], when the spatial distribution of particles is dense, it is the near-field TM storage that drives the storage requirement of the FFT T-matrix method. As shown in [12] and Appendix, $\alpha^{l,m}_{l',m'}(\bar{r}_j-\bar{r}_i)$ satisfies the following two symmetry relations involving the modal indices (l,m) and (l',m'):

$$\alpha^{l,m}_{l',m'}(\bar{r}_j-\bar{r}_i)=(-1)^{l+l'}e^{2i(m-m')\phi_{ji}}\alpha^{l',m'}_{l,m}(\bar{r}_j-\bar{r}_i) \tag{3.4}$$

and

$$\alpha^{l,-m}_{l',-m'}(\bar{r}_j-\bar{r}_i)=(-1)^{m+m'}e^{2i(m-m')\phi_{ji}}\alpha^{l,m}_{l',m'}(\bar{r}_j-\bar{r}_i), \tag{3.5}$$

where ϕ_{ji} is the spherical azimuth angle of $\bar{r}_j-\bar{r}_i$. Since particle i is in the near field of particle j if particle j is in the near field of particle i, the following parity-symmetry relation can also be used to reduce the memory requirement:

$$\alpha^{l,m}_{l',m'}(\bar{r}_i-\bar{r}_j)=(-1)^{l+l'}\alpha^{l,m}_{l',m'}(\bar{r}_j-\bar{r}_i). \tag{3.6}$$

Overall, (3.4)-(3.6) together reduce the storage requirement of the near-field TM by a factor of nearly 8. (It is not exactly 8 since some modal combinations, such as $\alpha^{1,-1}_{1,-1}$, can be double counted by (3.4) and (3.6)).

When the particle distribution is not too dense (but dense enough so that the FFT-based convolution technique can be used profitably), it is the storage of the far-field TM, $\tilde{\alpha}^{l,m}_{l',m'}(\bar{q})$, that determines the overall storage requirement of the FFT T-Matrix method. Since all $\bar{q}\equiv(q_x,q_y,q_z)$ values reside on an extended regular Cartesian grid, their reflections about the q_xq_y-, q_yq_z- and q_zq_x-planes also reside on the same regular grid. Therefore, the following reflection-symmetry relations can be used to reduce the far-field TM storage:

$$\begin{aligned}
&\tilde{\alpha}^{l,m}_{l',m'}(-q_x,q_y,q_z)=\tilde{\alpha}^{l,-m}_{l',-m'}(q_x,q_y,q_z),\\
&\tilde{\alpha}^{l,m}_{l',m'}(q_x,-q_y,q_z)=(-1)^{m+m'}\tilde{\alpha}^{l,-m}_{l',-m'}(q_x,q_y,q_z),\\
&\tilde{\alpha}^{l,m}_{l',m'}(q_x,q_y,-q_z)=(-1)^{l+m+l'+m'}\tilde{\alpha}^{l,-m}_{l',-m'}(q_x,q_y,q_z),\ etc.
\end{aligned} \tag{3.7}$$

With the above reflection symmetry relations, we need to store only $\tilde{\alpha}^{l,m}_{l',m'}(q_x,q_y,q_z)$ with

$$q_x\ge 0,\quad q_y\ge 0,\quad q_z\ge 0,$$

thus reducing the storage requirement of the far-field TM by a factor of 8. (3.4) and (3.5) individually are ill-suited for the FFT-based convolution due to the presence of the factor $e^{2\imath(m-m')\phi_{ji}}$. However, we may combine the two equations and obtain

$$\tilde{\alpha}^{l,-m}_{l',-m'}(\bar{q})=(-1)^{l+m+l'+m'}\tilde{\alpha}^{l',m'}_{l,m}(\bar{q}), \tag{3.8}$$

which permits an additional reduction by a factor of nearly two. Thus, we achieve a total storage reduction of the far-field TM by a factor of nearly 16 with (3.7) and (3.8).

Similarly, we may apply (3.4)-(3.6) to reduce the memory requirements of $\beta^{L',M'}_{l',m'}(\bar{d}_j)$ and $\beta^{l,m}_{L,M}(-\bar{d}_i)$ in (3.3). However, the magnitude of the resulting savings is small compared with those of $\alpha^{l,m}_{l',m'}(\bar{r}_j-\bar{r}_i),\ r_{ij}<\delta$ and $\tilde{\alpha}^{l,m}_{l',m'}(\bar{q})$, and thus we do not pursue their memory reduction in this work.

With (3.4)-(3.8), we need to compute and store only the minimum sets containing the independent elements of the configuration-space near-field TM and the k-space far-field TM and generate missing elements from them on the fly as they are needed. The computational cost of generating the missing elements of the near- and far-field TMs using the symmetry relations scales linearly with N_p and N_G, respectively. Since the overall CPU requirement scales as $\mathcal{O}(N_G\log_2 N_G)$ when N_p is large, it is expected that the relative cost of executing the symmetry operations decreases as the number of unknowns increases.

4 FFT T-matrix method with diagonal translator

One potentially serious drawback of the FFT T-matrix method discussed in the previous section is that both $\alpha^{l,m}_{l',m'}(\bar{r}_j-\bar{r}_i),\ r_{ji}<\delta$ and $\tilde{\alpha}^{l,m}_{l',m'}(\bar{q})$ are non-diagonal. That is, they depend on both (l,m) and (l',m'). As a consequence, even though the cost of performing the convolution, (3.2), is $\mathcal{O}({L_T}^2 N_G\log_2 N_G)$ as discussed in the previous section, it still requires, respectively, $\mathcal{O}({L_T}^4 N_G)$ and $\mathcal{O}({L_{ex}}^4 N_p)$ MUs to store the far-field $\tilde{\alpha}^{l,m}_{l',m'}(\bar{q})$ and the near-field $\alpha^{l,m}_{l',m'}(\bar{r}_j-\bar{r}_i)$, $r_{ji}<\delta$ that are needed to compute $\Gamma^{(F)}_{l',m'}(\bar{r}_j)$ and $\Gamma^{(N)}_{l',m'}(\bar{r}_j)$ in (3.2). [11] was able to reduce the far-field TM storage requirement from $\mathcal{O}({L_T}^4 N_G)$ to $\mathcal{O}({L_T}^2 N_G)$ by expanding the far-field $\alpha^{l,m}_{l',m'}(\bar{r}_j-\bar{r}_i)$, $r_{ji}>\delta$, in terms of planes waves and the diagonal translation operator [10]. In this section we examine a set of reflection symmetry relations of the diagonal translation operator that are particularly well suited for the FFT-based fast-convolution technique and show that they reduce the far-field TM storage requirement by an additional factor of 8.

4.1 Original formulation

As shown in Fig. 1, we let $\bar{r}_i=\bar{R}_i+\bar{d}_i$ and $\bar{r}_j=\bar{R}_j+\bar{d}_j$, where $\bar{R}_i$ and $\bar{R}_j$ are the grid points nearest to $\bar{r}_i$ and $\bar{r}_j$, respectively. The far-field $\alpha^{l,m}_{l',m'}(\bar{r}_j-\bar{r}_i)$, $r_{ji}>\delta$, may be expanded in terms of plane waves [11]:

$$\alpha^{l,m}_{l',m'}(\bar{r}_j-\bar{r}_i)=\int_{S_k} e^{i\bar{k}\cdot\bar{d}_j} i^{-l} Y_{l,m}(\hat{k})\tau_{L_T}(\hat{k},\bar{R}_j-\bar{R}_i) i^{l'} Y^*_{l',m'}(\hat{k}) e^{-i\bar{k}\cdot\bar{d}_i} d\hat{k}, \tag{4.1}$$

where

$$\hat{k}\equiv(\theta_k,\phi_k),\quad \int_{S_k}(\cdot)d\hat{k}\equiv\int_{-1}^{1}\int_0^{2\pi}(\cdot)d(\cos\theta_k)d\phi_k.$$

$\tau_{L_T}(\hat{k},\bar{R}_j-\bar{R}_i)$ is the diagonal translator [10],

$$\tau_{L_T}(\hat{k},\bar{R}_j-\bar{R}_i)\equiv\sum_{l=0}^{L_T} i^l(2l+1)h_l(kR_{ji})P_l(\hat{k}\cdot\hat{R}_{ji}), \tag{4.2}$$

where $\bar{R}_{ji}\equiv\bar{R}_j-\bar{R}_i$; $h_l(.)$ is the spherical Hankel function of order l; $P_l(.)$, the Legendre polynomial of order l; and $\hat{R}_{ji}\equiv\bar{R}_{ji}/R_{ji}$. The numerical integration over $\hat{k}$ in (4.1) requires (L_T+1) Gauss-Legendre quadrature points for the θ_k integration and $2(L_T+1)$ trapezoidal points for the ϕ_k integration [11]. Thus, a total of $2(L_T+1)^2$ quadrature points are needed to perform the $\hat{k}$ integration numerically. Since $\tau_{L_T}(\hat{k},\bar{R}_{ji})$ is doubly-block Toeplitz for given $\hat{k}$, the far-field convolution, $\Gamma^{(F)}_{l',m'}(\bar{r}_j)$, in (3.2) may be written as

$$\begin{aligned}&\sum_{l,m}\sum_{i\in\mathcal{F}_j}\alpha^{l,m}_{l',m'}(\bar{r}_j-\bar{r}_i)\cdot b^{(i)}_{l,m}\\
&=\sum_{p=1}^{N_k} w_p e^{i\bar{k}_p\cdot\bar{d}_j} i^{-l}Y_{l,m}(\hat{k}_p)\sum_i \hat{\tau}_{L_T}(\hat{k}_p,\bar{R}_{ji})\, i^{l'} e^{-i\bar{k}_p\cdot\bar{d}_i}\sum_{l,m} Y^*_{l',m'}(\hat{k}_p)b^{(i)}_{l,m}\\
&=\sum_{p=1}^{N_k} w_p e^{i\bar{k}_p\cdot\bar{d}_j} i^{-l}Y_{l,m}(\hat{k}_p)\,\mathrm{FFT}^{-1}\left[\tilde{\tau}_{L_T}(\hat{k}_p,\bar{q})\cdot\tilde{\chi}(\hat{k}_p,\bar{q})\right].\end{aligned} \tag{4.3}$$

Here, $\hat{k}_p$ and w_p, respectively, represent the quadrature points and weights and $N_k\equiv 2(L_T+1)^2$. $\tilde{\tau}_{L_T}(\hat{k}_p,\bar{q})$ and $\tilde{\chi}(\hat{k}_p,\bar{q})$, respectively, are the Fourier transforms of $\hat{\tau}_{L_T}(\hat{k}_p,\bar{R}_{ji})$ and $\chi(\hat{k}_p,i)$, where

$$\hat{\tau}_{L_T}(\hat{k}_p,\bar{R}_{ji})\equiv\begin{cases}\tau_{L_T}(\hat{k}_p,\bar{R}_{ji}) & \text{if } |\bar{R}_j-\bar{R}_i|>\delta,\\ 0 & \text{otherwise,}\end{cases}$$

and

$$\chi(\hat{k}_p,i)\equiv e^{-i\bar{k}_p\cdot\bar{d}_i}\sum_{l,m} i^{-l}Y^*_{l,m}(\hat{k}_p)b^{(i)}_{l,m}.$$

In (4.3), $\tilde{\tau}_{L_T}(\hat{k},\bar{q})$ needs to be computed only once at the cost of $\mathcal{O}({L_T}^2 N_G\log_2 N_G)$ FPOs and requires $\mathcal{O}({L_T}^2 N_G)$ MUs to store them. Thus, the use of the diagonal translator reduces the memory requirement of the far-field TM from $\mathcal{O}({L_T}^4 N_G)$ to $\mathcal{O}({L_T}^2 N_G)$.

Since (4.1) is not valid when $r_{ji}\le\delta$, the memory requirement of the near-field TM remains unchanged (if the same δ is used). As noted in [13], when the particle distribution is dense, it is the memory requirement of the near-field TM that drives the overall memory requirements, and thus (4.1) provides only a marginal improvement over the FFT T-matrix method based on the non-diagonal translator. However, as the particle distribution becomes more inhomogeneous or tenuous, it may provide a substantial improvement over the non-diagonal translator-based method, since for this type of particle distribution it is the far-field TM memory requirement that drives the overall memory requirements. The computation of (4.3) requires $2(L_T+1)^2$ forward and $2(L_T+1)^2$ inverse

FFT operations, while (3.3), as discussed in Section 3.1, requires $(L_T+1)^2$ forward and $(L_T+1)^2$ inverse FFT operations. Therefore, the FFT T-matrix method based on the diagonal translator requires twice as many spatial convolutions for a comparable numerical accuracy.

4.2 Memory requirement reduction

Since it is only the far-field TM that gets diagonalized, the memory requirement of the near-field TM remains the same as that of the FFT T-matrix method discussed in the previous section (if the same δ value is used). Thus with (3.4)-(3.6), as discussed in Section 3.2, the memory requirement of the near-field TM can be reduced by a factor of nearly eight.

It is also possible to further reduce the memory requirement of the far-field TM, $\tilde{\tau}_{L_T}(\hat{k},\bar{q})$. The scalar product, $\hat{k}\cdot\bar{R}_{ji}$, in (4.2) is invariant under the simultaneous reflections of R_{ji} about the xy-, yz-, and zx-planes and $\hat{k}$ about the k_xk_y-, k_yk_z-, and k_zk_x-planes. Therefore, if we let $\hat{k}\equiv(\hat{k}_x,\hat{k}_y,\hat{k}_z)$ and $\bar{r}\equiv(x,y,z)$, the following set of eight reflection symmetry relations of $\tau_{L_T}(\hat{k},\bar{r})$ results:

$$\begin{aligned}&\tau_{L_T}(\hat{k},\bar{r})\equiv\tau_{L_T}(\hat{k}_x,\hat{k}_y,\hat{k}_z;x,y,z)=\tau_{L_T}(-\hat{k}_x,\hat{k}_y,\hat{k}_z;-x,y,z)\\ =&\tau_{L_T}(\hat{k}_x,-\hat{k}_y,\hat{k}_z;x,-y,z)=\tau_{L_T}(\hat{k}_x,\hat{k}_y,-\hat{k}_z;x,y,-z)\\ =&\tau_{L_T}(-\hat{k}_x,-\hat{k}_y,\hat{k}_z;-x,-y,z)=\tau_{L_T}(\hat{k}_x,-\hat{k}_y,-\hat{k}_z;x,-y,-z)\\ =&\tau_{L_T}(-\hat{k}_x,\hat{k}_y,-\hat{k}_z;-x,y,-z)=\tau_{L_T}(-\hat{k}_x,-\hat{k}_y,-\hat{k}_z;-x,-y,-z).\end{aligned}\tag{4.4}$$

The $\bar{R}_{ji}$ in (4.2) reside on an extended regular Cartesian grid (i.e., for each $\hat{k}_p$, $\hat{\tau}_{L_T}(\hat{k}_p,\bar{R}_{ji})$ is doubly-block circulant) and their reflection points about the xy-, yz- and zx-planes also reside on the same extended regular grid. The above eight reflection symmetry relations also hold for $\tilde{\tau}_{L_T}(\hat{k},\bar{q})$ in the Fourier domain:

$$\begin{aligned}&\tilde{\tau}_{L_T}(\hat{k},\bar{q})\equiv\tau_{L_T}(\hat{k}_x,\hat{k}_y,\hat{k}_z;q_x,q_y,q_z)=\tau_{L_T}(-\hat{k}_x,\hat{k}_y,\hat{k}_z;-q_x,q_y,q_z)\\ =&\tau_{L_T}(\hat{k}_x,-\hat{k}_y,\hat{k}_z;q_x,-q_y,q_z)=\tau_{L_T}(\hat{k}_x,\hat{k}_y,-\hat{k}_z;q_x,q_y,-q_z)\\ =&\tau_{L_T}(-\hat{k}_x,-\hat{k}_y,\hat{k}_z;-q_x,-q_y,q_z)=\tau_{L_T}(\hat{k}_x,-\hat{k}_y,-\hat{k}_z;q_x,-q_y,-q_z)\\ =&\tau_{L_T}(-\hat{k}_x,\hat{k}_y,-\hat{k}_z;-q_x,q_y,-q_z)=\tau_{L_T}(-\hat{k}_x,-\hat{k}_y,-\hat{k}_z;-q_x,-q_y,-q_z).\end{aligned}\tag{4.5}$$

Since $\bar{q}\equiv(q_x,q_y,q_z)$ reside on the extended regular Cartesian grid in the Fourier space, so do their reflection points. The aforementioned optimum $\hat{k}$-integration scheme, based on the Gauss-Legendre quadrature method for the θ_k integration and the trapezoidal rule for the ϕ_k integration, also produces quadrature points and weights that remain symmetric under the reflections. Therefore, when the symmetry properties of the quadrature points and their weights are combined with (4.5), it is necessary to compute and store only those $\tilde{\tau}_{L_T}(\bar{k},\bar{q})$ values that correspond to the first octant of $\bar{q}$ with $q_x\geq 0$, $q_y\geq 0$, and $q_z\geq 0$. Thus, the far-field TM memory requirement is reduced by a factor of eight.

5 Validation and CPU comparisons

In this section we validate the memory- and convolution-reduction schemes introduced in the previous sections, compare the CPU times required to perform the matrix-vector multiplication, (3.2), by the two FFT-based fast-convolution methods, and show that the relative cost of using the symmetry relations decreases as N increases.

For the validation, we compute the acoustic bistatic scattering cross section of an ensemble of 100 identical $0.1\lambda_0$-diameter spheres that are randomly distributed inside a $(1.5\lambda_0)^3$ cubic box using three different techniques[‡]: (i) the standard conjugate-gradient (CG) method without FFT acceleration, (ii) the CG-FFT method with the non-diagonal translator, and (iii) the CG-FFT method with the diagonal translator. When using the CG-FFT method with the non-diagonal translator, we use (3.4)-(3.6) to reduce the near-field TM storage requirement, and (3.7) and (3.8) to reduce the far-field TM storage requirement. When computing the far-field convolution, (3.3), we perform the spatial convolution $(L_T+1)^2$ times. Similarly, when using the CG-FFT method with the diagonal translator, we use (3.4)-(3.6) to reduce the near-field TM storage requirement and (4.5) to reduce the far-field TM storage requirement. The result obtained using the standard CG method without FFT acceleration serves as a reference solution.

For all three techniques, we use 9 partial waves corresponding to $L_{ex}=2$ in (2.1) (thus there are 900 unknown $b_{l,m}^{(j)}$ coefficients) and terminate the CG iteration when the normalized residual error (NRE) reaches 0.001. When translating $\alpha_{l',m'}^{l,m}(\bar{r}_j-\bar{r}_i)$ for the CG-FFT method with the non-diagonal translator, we also retain 9 partial waves corresponding to $L_T=2$ in (3.1). For the FFT method with the diagonal translator, we use 18 quadrature points (corresponding to 3 and 6 points for the θ_k and ϕ_k integrations, respectively) to perform the $\hat{k}$ integration in (4.1). For the two CG-FFT methods, we use the forward and inverse FFTs of size 32 x 32 x 32 that extends from $-1.5\lambda_0$ to $1.5\lambda_0$ in each Cartesian direction with the grid spacing $d=0.1\lambda_0$ (λ_o is the wavelength). Thus, in (3.1),

$$\max(d_i,d_j)=(\sqrt{3}/2)d\simeq 0.086\lambda_0.$$

When using the standard CG method without FFT acceleration, we need to store a complex matrix of size 900 by 900. For both CG-FFT methods without memory reduction, we need approximately $81\bar{N}_n N_p$ complex words (CWs) to store the near-field TM, where 81 is the total number of (l,m) and (l',m') combinations corresponding to $L_{ex}=2$, N_p the number of particles in the ensemble and $\bar{N}_n$ the average number of particles in the near field. $\bar{N}_n$ is equal to the product of the particle density and the near-field volume. With $\delta=0.4\lambda_0$ in (3.2), we obtain $\bar{N}_n\approx 7.9$. Thus, for both CG-FFT methods without memory reduction, we need approximately 6.4×10^4 CWs to store the near-field TM. The two modal symmetry relations, (3.4)-(3.5), reduce the number of (l,m) and (l',m') combinations from 81 to 24, while the parity-symmetry relation, (3.6), provides an additional

[‡]Admittedly, the particle density may not be high enough to warrant the use of the FFT-based techniques.

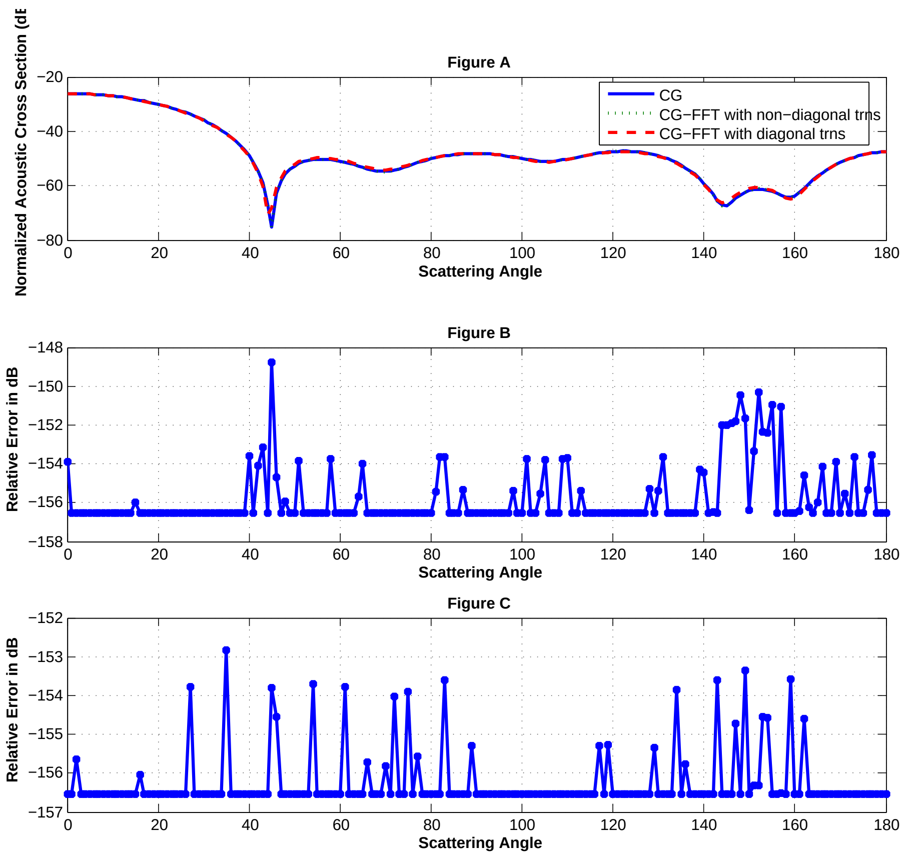

Figure 2: (A): Acoustic bistatic scattering cross section, as a function of scattering angle, of an ensemble of 100 $0.1\lambda_0$-diameter spheres that are randomly distributed inside a $(1.5\lambda_0)^3$ cubic volume, computed using the standard CG method without FFT acceleration (solid); the CG-FFT method with non-diagonal TM (dotted); and CG-FFT with diagonal TM(dashed). (B): Relative error in dB between the original CG-FFT method based on non-diagonal TM (without memory and convolution reductions) and the improved CG-FFT based on non-diagonal TM with convolution and memory reductions. (C): Relative error in dB between the original CG-FFT based on diagonal TM (without memory reduction) and the improved CG-FFT based on diagonal TM with memory reduction.

twofold reduction. Thus, the near-field TM can be stored using only $\sim 9.5\times 10^3$ CWs. For the far-field TM storage, the original CG-FFT method with the non-diagonal translator requires $81\times 32^3 \approx 2.7\times 10^6$ CWs, while the original CG-FFT method with the diagonal translator needs only $18\times 32^3 \approx 5.9\times 10^5$ CWs. (3.8) reduces the number of independent (l,m) and (l',m') combinations from 81 to 45, while (3.7) produces an eightfold reduction. Thus, the far-field non-diagonal TM requires only $45\times 16^3 \approx 1.8\times 10^5$ CWs. Similarly, (4.5) reduces the storage for the diagonal far-field TM to $18\times 16^3 \approx 7.4\times 10^4$ CWs. We note that, even for this low-density particle distribution, the storage requirements of the two CG-FFT methods, augmented with the symmetry relations, are milder than that of the standard CG method without FFT acceleration.

Compared in Fig. 2(A) are the three sets of bistatic cross section values predicted by the three techniques mentioned above. As evident from the figure, they are in excellent agreement with each other. Fig. 2(B) plots the relative error in dB between the original FFT method with the non-diagonal translator (without the memory and convolution reductions) and the improved FFT method with the non-diagonal translator with the memory and convolution reductions. The relative error is defined as

$$\text{relative error in dB} = 10\log_{10}\left[\mathbf{max}\left(\left|\frac{\sigma_1(\theta)-\sigma_2(\theta)}{\sigma_1(\theta)}\right|, 2^{-52}\right)\right],$$

where $\sigma_1(\theta)$ and $\sigma_2(\theta)$ are the complex scattering amplitudes computed using method 1 and method 2, respectively. The presence of $2^{-52} \approx -156$ dB ensures that we don't take the logarithm of 0 when $\sigma_1(\theta) = \sigma_2(\theta)$. Similarly, Fig. 2(C) plots the relative error in dB between the original FFT method with the diagonal translator (without the memory reductions) and the improved FFT method (with the memory reductions). It is evident that neither the memory reduction nor the convolution reduction affects the accuracy of the CG-FFT methods.

In Fig. 3, we compare, as a function of N, the CPU times needed to execute the matrix-vector multiplication (2.2) by the brute-force method and (3.2) by the two FFT methods. As defined in Section 2, $N \equiv PN_p \equiv (L_{ex}+1)^2 N_p$ and we set $L_{ex} = 2$. For the two FFT methods, the figure shows the size of the regular grid for each N (the size of the extended regular grid, which corresponds to the actual size of the FFT and IFFT operations, is eight times the size of the regular grid). For the brute-force method, the CPU times for $N > 20000$ are estimated using a quadratic fit to the actual CPU data for $N < 20000$. Translation onto the regular grid is achieved using $L_T = 2$ for the FFT method with the non-diagonal translator and $N_{\theta_k} = 3$ and $N_{\phi_k} = 6$ for the FFT methods with the diagonal translator. For both FFT methods, we set $\delta = 0.3\lambda_0$ and the CPU time data for the two FFT methods are obtained without using the symmetry relations. As expected, the two FFT methods perform the matrix-vector multiplications significantly faster than the brute-force method as N increases. Since the FFT method with the non-diagonal translator requires half as many spatial convolutions as the FFT method based on the diagonal method, the figure shows that the former is about 15 $\sim$ 30 % more efficient than the latter.

The use of the symmetry relations, which reduces the memory requirements as discussed in the previous sections, will invariably increase the CPU time needed to perform matrix-vector multiplications. We note that the phase changes associated with (3.7) and (3.8) are separable in (l,m) and (l',m') and independent of $\bar{q}$, while (4.5) requires no phase change. The near-field TM symmetry relations, (3.4), (3.5), and (3.6), are used by both FFT methods. However, the phase changes associated with (3.4) and (3.5), unlike those accompanying (3.7), (3.8) and (4.5), are dependent on $e^{i2(m-m')\phi_{ij}}$ and thus (3.4) and (3.5) are computationally more expansive to use than (3.7), (3.8) and (4.5). In order not to increase the cost of generating missing near-field TM elements too much, we may substitute (3.4)

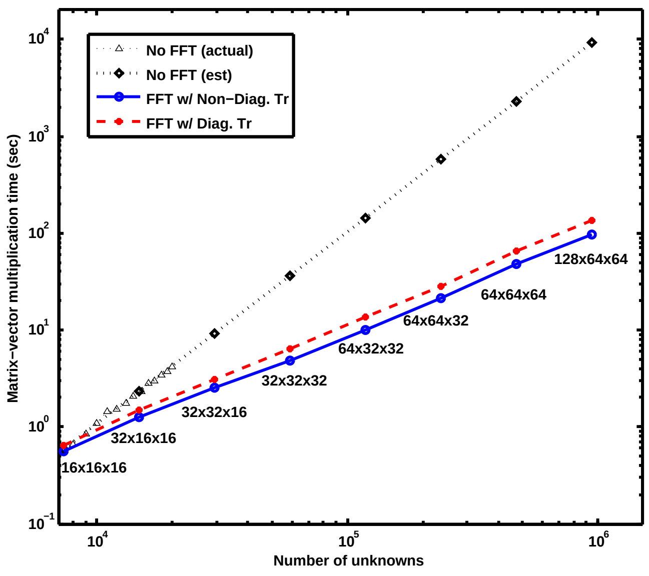

Figure 3: Comparison as a function of N of the CPU times required to perform the matrix-vector multiplication, (2.2) by the brute-force method (dotted) and (3.2) by the FFT methods based on the non-diagonal translator (solid) and the FFT method based the diagonal translator (dashed).

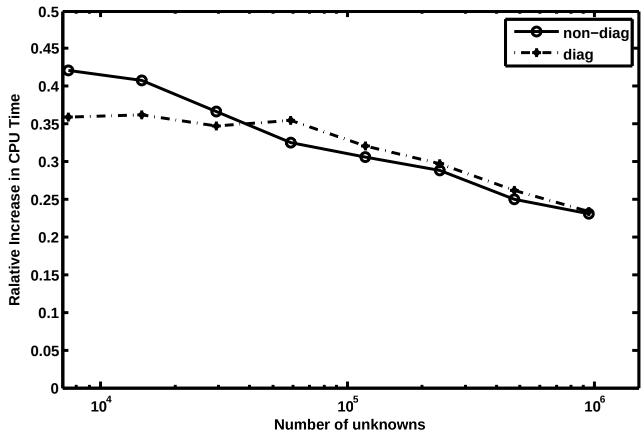

Figure 4: Relative cost, as a function of N, of using the symmetry relations to generate missing TM elements.

and (3.5) with the configuration-space equivalent of (3.8),

$$\alpha_{l',-m'}^{l,-m}(\bar{r}_i-\bar{r}_j)=(-1)^{l+m+l'+m'}\alpha_{l,m}^{l',m'}(\bar{r}_i-\bar{r}_j), \tag{5.1}$$

whose phase factor is independent of ϕ_{ij} and separable in (l,m) and (l',m'). The overall near-field memory requirement would then be reduced by a factor of four (rather than by a factor of 8).

Table 1: For the definitions of the expansion parameters L_{ex} and L_T, please see (2.1) and (3.1), respectively. N_s is the number of scatterers in the ensemble. * The near-field memory reduction for the timing data shown in Fig. 4 is $\sim 1/4(L_{ex}+1)^2 N_p$.

	Non-Diagonal Translation $\alpha_{l',m'}^{l,m}(\bar{r})$		Diagonal Translation $\tau_{L_T}(\hat{k},\bar{r})$	
	No Memory, No Convolution Reduction	Reduced Convolution & Reduced Memory	No Memory Reduction	Reduced Memory
# of Convolutions	$(L_T+1)^4$	$(L_T+1)^2$	$2(L_T+1)^2$	$2(L_T+1)^2$
Near-Field Memory	$(L_{ex}+1)^4 N_s$	$\sim 1/8\ (L_{ex}+1)^4 N_s{}^*$	$(L_{ex}+1)^4 N_s$	$\sim 1/8(L_{ex}+1)^4 N_s{}^*$
Far-Field Memory	$(L_T+1)^4 N_G$	$\sim 1/16\ (L_T+1)^4 N_G$	$2(L_T+1)^2 N_G$	$1/4(L_T+1)^2 N_G$

Fig. 4 plots the relative CPU time increases for the two FFT methods when the symmetry relations are employed to generate missing TM elements for the cases considered in Fig. 3. As the figure shows, the relative increase in CPU time decreases as N increases for both methods. Since the overall CPU time for the matrix-vector multiplication scales as $\mathcal{O}(N\log_2 N)$ while the cost of using the symmetry relations scales as $\mathcal{O}(N)$, the relative cost of using the symmetry relations becomes smaller as N increases.

6 Conclusion

We compared and improved the computational efficiencies of two existing FFT-based iterative-solution techniques for the scalar T-matrix multiple-scattering equation. We showed that the FFT method based on the non-diagonal translator requires half as many spatial convolutions as the FFT method based on the diagonal translator for a comparable numerical accuracy. The CPU time comparison shown in Fig. 3 confirms that the first method requires less CPU time to perform matrix-vector multiplications. When solving scattering from an ensemble containing a large number of scatterers, it is often the memory requirement that determines the largest problem size that can be solved on a given computer. For both FFT methods, storage requirements for the configuration-space near-field TM and the k-space far-field TM drive the overall storage requirements. We identified a number of symmetry relations that the non-diagonal and diagonal TMs satisfy in configuration- and k-spaces. These symmetry relations are shown to reduce the overall storage requirements by non-trivial factors. We also showed that the relative cost of using the symmetry relations becomes smaller as the number of unknowns increases. Table 1 summarizes the memory and convolution reductions discussed in this paper. They can easily be implemented in an existing multiple-scattering T-matrix code, allowing it solve larger problems faster on a given computer.

Appendix: Proof of (3.4)

In this appendix, we provide a derivation of (3.4). All other symmetry relations of $\alpha^{l,m}_{l',m'}(\bar{r})$ used in this paper are derived in [12].

Following [9], we may write

$$\alpha^{l,m}_{l',m'}(\bar{r}) = \sum_{L=|l-l'|,2}^{l+l'} \Lambda^{l,m}_{l',m'}(L)\cdot h_L(kr) Y_{L,m-m'}(\hat{r}), \tag{A.1}$$

with

$$\Lambda^{l,m}_{l',m'}(L) \equiv 4\pi i^{l'-l-L}(-1)^{m'}\sqrt{\frac{(2l+1)(2l'+1)}{4\pi(2L+1)}}\mathcal{C}(l,l',L;0,0,0)\mathcal{C}(l,l',L;-m,m',-m+m'). \tag{A.2}$$

Here, $h_L(kr)$ is the spherical Hankel function of order L; $Y_{L,m-m'}(\hat{r})$, the spherical harmonics of order $(L,m-m')$; $\bar{r}\equiv(r,\theta,\phi)$; $\hat{r}\equiv(\theta,\phi)$, and $\mathcal{C}(a,b,c;m_a,m_b,m_c)$, the Clebsch-Gordan coefficient.

To prove (3.4), it is necessary to relate $\Lambda^{l,m}_{l',m'}(L)$ to $\Lambda^{l',m'}_{l,m}(L)$ and $Y_{L,m'-m}(\hat{r})$ to $Y_{L,m-m'}(\hat{r})$. Since $\mathcal{C}(l,l',L;0,0,0)\neq 0$ only when $l+l'+L$ is even in (A.2), and since

$$\mathcal{C}(b,a,c;m_b,m_a,m_c) = (-1)^{a+b+m_b+m_c}\mathcal{C}(a,b,c;m_a,m_b,m_c), \tag{A.3}$$

$$\mathcal{C}(a,b,c;-m_b,-m_a,-m_c) = (-1)^{a+b+m_b+m_c}\mathcal{C}(a,b,c;m_a,m_b,m_c), \tag{A.4}$$

one can show that

$$\Lambda^{l,m}_{l',m'}(L) = (-1)^{l+l'+m+m'}\Lambda^{l',m'}_{l,m}(L). \tag{A.5}$$

In addition, using the following property of the associated Legendre polynomial,

$$P_l^{-m}(\cos\theta) = (-1)^m\frac{(l-m)!}{(l+m)!}P_l^m(\cos\theta),$$

one can show that

$$Y_{l,-m}(\hat{r}) = (-1)^m e^{-2im\phi}Y_{l,m}(\hat{r}). \tag{A.6}$$

Thus, after substituting (A.5) and (A.6) into (A.1), we have

$$\begin{aligned}\alpha^{l,m}_{l',m'}(\bar{r}) &= \sum_{L=|l-l'|,2}^{l+l'} \Lambda^{l,m}_{l',m'}(L)\cdot h_L(kr)Y_{L,m-m'}(\hat{r})\\ &= \sum_{L=|l-l'|,2}^{l+l'}(-1)^{l+l'+m+m'}\Lambda^{l',m'}_{l,m}(L)\cdot(-1)^{m+m'}e^{2i(m-m')\phi}h_L(kr)Y_{l,m'-m}(\hat{r})\\ &= (-1)^{l+l'}e^{2i(m-m')\phi}\alpha^{l',m'}_{l,m}(\bar{r}),\end{aligned}$$

which proves (3.4).

References

[1] B. Peterson and S. Ström, T Matrix for Electromagnetic Scattering from an Arbitrary Number of Scatterers and Representation of E(3), *Physical Review D*, Vol. 8, No, 10, 3661-3678, 1973.

[2] P.C. Waterman, Matrix Formulation of Electromagnetic Scattering, *Proceedings of the IEEE*, Vol. 53, No. 8, 805-812, 1965.

[3] G.H. Goub and C.F. Van Loan, *Matrix Computations*, 2nd Edition. The Johns Hopkins University Press, Baltimore, MD, 1989.

[4] M.R. Hestenes and E. Stiefel, Methods of Conjugate Gradients for Solving Linear Systems, *J. Res. Nat. Bur. Stad.*, Vol. 39, pp. 409-435, 1952.

[5] D.J. Jones, *Acoustic and Electromagnetic Waves*, Oxford University Press, New York, NY, 1986.

[6] C.H. Chan and L. Tsang, A Sparse-Matrix Canonical-Grid Method for Scattering by Many Scatterers, *Microwave Opt. Technol. Lett.*, Vol. 8, No. 2, pp. 114-118, 1995.

[7] W.C. Chew, J.H. Lin and X.G. Yang, An FFT T-Matrix Method for 3D Microwave Scattering Solutions from Random Discrete Scatterers, *Microwave. Opt. Technol. Lett.*, Vol. 9, No. 4, 194-196, 1995.

[8] M. Frigo and S.G. Johnson, The Design and Implementation of FFTW3, *Proc. IEEE*, Vol. 93, No. 2, pp. 216-231, 2005.

[9] W.C. Chew, *Waves and Fields in Imhomogeneous Media,* IEEE Press, New Jersey, 1995.

[10] R. Coifman, V. Rokhlin, and S. Wandzura, The Fast Multipole Method for the Wave Euqation, *IEEE Trans. Antennas Propagat. Mag.* Vol.35, No. 3, 7-12, 1993.

[11] S. Koc and W.C. Chew, Culculation of Acoustic Scattering from a Cluster of Scatterers, *Journal of Acoustical Society of America*, Vol. 103, No. 2, pp 721-734, 1998.

[12] K.T. Kim, The Symmtery Relations of the Translation Coefficients of the Scalar and Vector Spherical Multipole Fields, *Progress in Electromagntic Research, PIER 48*, Chapter 3, pp. 45-66, 2004.

[13] K.T. Kim, A Memory-Reduction Scheme for the FFT T-Matrix Method, *IEEE Wireless Antenna and Propagat. Letters*, Vol. 3, No 10, pp. 193-196, 2004.