

MSTM

A multiple sphere T -matrix FORTRAN code for use on parallel
computer clusters

Version 2.2

D. W. Mackowski
Department of Mechanical Engineering
Auburn University, Auburn, AL 36849, USA
mackodw@auburn.edu

11 November 2011

About this document

This is the instruction manual for the MSTM Fortran-90 code. The code was originally released in January 2011. Code elements will be revised in response to bug fixes, user suggestions, and modifications/extensions; the revision date will appear in comment lines at the top of each module and/or subroutine of the code.

The current version of the manual (the one you are reading now) corresponds to version 2.2, and was released on 11 November 2011. The revision history appears in [Sec. \(6\)](#).

Michael Mishchenko has contributed significantly both to the development of the code and its visibility. Development has also benefitted from the advice and encouragement provided, over the years, by Zhanna Dlugach, Bruce Draine, Piotr Flatau, Kirk Fuller, Joop Hovenier, Michael Kahnert, Nikolai Khlebtsov, Ludmilla Kolokolova, Li Liu, Pinar Menguç, George Mulholland, Olga Munoz, Antti Pentillä, Jani Tyynelä, Michael Wolff, Thomas Wriedt, and Evgenij Zubko.

All queries regarding the code should be addressed to the author at mackodw (at) auburn.edu

Contents

1 Purpose	4
2 Mathematical Formulation	4
2.1 Interaction equations	4
2.2 Numerical solution of the interaction equations	6
2.3 Incident and total scattered field	8
2.4 Coordinate rotation, amplitude and scattering matrix, and cross sections	10
2.5 The T matrix relationships	12
2.6 Random orientation	13
3 Multiple sphere T-matrix (MSTM) code	14
3.1 Structure and compilation	14
3.2 Prepackaged main program	14
3.2.1 Options related to sphere target specification	15
3.2.2 Options related to numerical solution	16
3.2.3 Options related to incident field and calculated quantities	17
3.2.4 Options for fixed orientation calculations	18
3.2.5 Options for random orientation calculations	19
3.2.6 Termination of input data	20
3.3 Parallel considerations	20
3.4 Output	20
3.4.1 Run (intermediate) print file	20
3.4.2 Output file	21
3.4.3 Near field file	21
4 Examples	22
4.1 Calculation of random orientation properties for a spherical cluster	22
4.2 Electric field distribution in a slab of spheres	25
5 Code elements and customization	31
5.1 Standard main program structure	31
5.2 MPI alias instructions	37
5.3 Code modification and customizing	39
6 Revisions	39
1.2 (21 February 2011)	39
2.1 (30 March 2011)	39
2.2 (11 November 2011)	39
Appendix	40
Optically active formulation	40
Vector Spherical Wave Functions	42
Wigner D functions	43
Translation matrix elements	44

Recurrence formula for axial translation	44
Vector spherical harmonics	45

1 Purpose

MSTM is a FORTRAN-90 code for calculating the time-harmonic electromagnetic scattering properties of a group of spheres. The algorithm applies the multiple sphere T matrix method, and the results can be considered exact to the truncation error of the vector spherical wave function (VSWF) expansions used to represent the fields. The code can

- calculate the cross sections, asymmetry parameters, and far-field scattering matrix elements for both fixed and random orientations with respect to the incident wave,
- model both plane wave and Gaussian profile incident beams, and
- generate maps of the electric field distributions along any arbitrary plane and including points both within and external to the spheres.

The MSTM code is intended to replace the FORTRAN-77 `scsmtm.for` and `sksmfo.for` codes that were previously developed by the author. In revising the multiple sphere scattering codes, the programming goals were to develop a code which

- is as compiler- and machine-independent as possible,
- can be compiled and run on both serial and distributed-memory parallel processing platforms,
- optimally uses the memory and (for parallel platforms) processor resources of the machine, and
- allows for a wide range of calculation and output options without modification and recompilation of the code.

Those familiar with the old codes should have little difficulty working with MSTM. The new code replaces all static array dimensions with dynamic memory allocation, and this eliminates the need to adjust `PARAMETER` statements to meet the memory requirements of the machine and/or the calculation. The code also incorporates message passing interface (MPI) commands to implement execution on distributed memory, multiple processor compute clusters.

A summary of the mathematical formulation and algorithm is given in [Sec. \(2\)](#), and compilation and execution of the code are described in [Sec. \(3\)](#). An overview of the code structure, and directions on how to modify the code for specialized applications, is found in [Sec. \(5\)](#).

2 Mathematical Formulation

2.1 Interaction equations

In the most general sense, the purpose of the code is to render a complete description of the electromagnetic fields, in both the near and far field regions, that result from the excitation of a target of N_S spheres with a time-harmonic field. A target, illustrated in [Fig. 1](#), is specified by the size parameters $x_i = ka_i = 2\pi a_i/\lambda$, relative bulk refractive indices $\mathbf{m}_i = \mathbf{m}'_i + i\mathbf{m}''_i$, where $i = \sqrt{-1}$, and positions relative to a common target origin $\mathbf{r}_i = (X_i, Y_i, Z_i)$ for $i = 1, 2, \dots, N_S$. The spheres can also be optically active, with distinct left and right-handed refractive indices (new in version 2.1). The optical activity of a sphere is determined by a

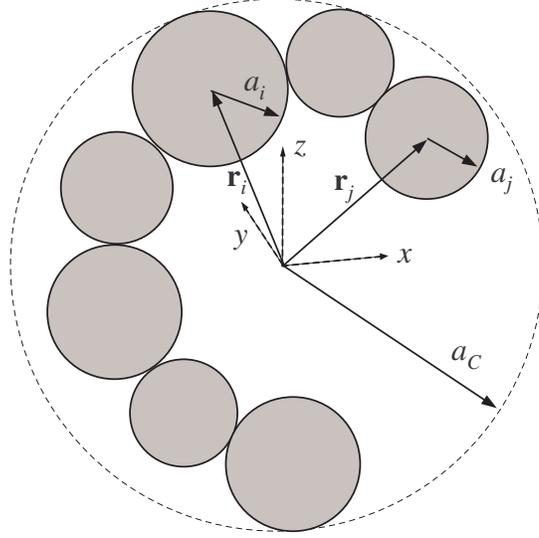


Figure 1: Ensemble configuration

complex chiral factor for the sphere, denoted as χ_i , so that the left and right hand refractive indices are given by

$$\mathbf{m}_L = \frac{\mathbf{m}}{1 - \chi \mathbf{m}}, \quad \mathbf{m}_R = \frac{\mathbf{m}}{1 + \chi \mathbf{m}}, \quad \mathbf{m} = \frac{2\mathbf{m}_L \mathbf{m}_R}{\mathbf{m}_L + \mathbf{m}_R} \quad (1)$$

The spheres can be in point contact with their neighbors, yet they cannot overlap.

The formulation represents an extension of Lorenz-Mie theory (for optically active spheres) to the multiple sphere system. The field external to the spheres is represented by the superposition of the incident and scattered fields, except in this case the scattered field consists of components radiated from each sphere in the target;

$$\mathbf{E}_{ext} = \mathbf{E}_{inc} + \mathbf{E}_{sca} = \mathbf{E}_{inc} + \sum_{i=1}^{N_S} \mathbf{E}_{sca,i} \quad (2)$$

The incident and scattered fields, at the i^{th} sphere in the cluster, can be represented by regular and outgoing vector spherical wave function (VSWF) expansions, centered about the origin of the sphere;

$$\mathbf{E}_{inc} = \sum_{n=1}^{L_i} \sum_{m=-n}^n \sum_{p=1}^2 J_{mnp}^i \mathbf{N}_{mnp}^{(1)}(\mathbf{r} - \mathbf{r}_i) \quad (3)$$

$$\mathbf{E}_{sca,i} = \sum_{n=1}^{L_i} \sum_{m=-n}^n \sum_{p=1}^2 a_{mnp}^i \mathbf{N}_{mnp}^{(3)}(\mathbf{r} - \mathbf{r}_i) \quad (4)$$

In the above, \mathbf{N}_{mnp} denotes the VSWF of either type 1 (regular) or 3 (outgoing), of order n , degree m , and mode $p = 1$ (TM) or 2 (TE). The functions are defined in the [Appendix](#). The incident field coefficients

f_{mnp}^i will depend on the characteristics of the incident field, whereas the scattered field coefficients a_{mnp}^i are unknown and are sought from the solution. The order truncation limit L_i in Eq. (4) is chosen to provide an acceptable level of convergence of the series; this topic will be covered in more detail below.

Application of the continuity equations at the surface of each sphere results in a system of interaction equations for the scattered field coefficients;

$$a_{mnp}^i - \sum_{p'=1}^2 \bar{a}_{npp'}^i \sum_{\substack{j=1 \\ j \neq i}}^{N_S} \sum_{l=1}^{L_j} \sum_{k=-l}^l \sum_{q=1}^2 H_{mnp'klq}^{i-j} \alpha_{klq}^j = \sum_{p'=1}^2 \bar{a}_{npp'}^i f_{mnp'}^i \quad (5)$$

in which $\bar{a}_{npp'}^i$ denote the optically active (OA) Mie coefficients of the sphere and depend on the sphere size parameter and L and R refractive indices, and H^{i-j} is the outgoing VSWF translation matrix which transforms the outgoing VSWF centered about origin j into an expansion of regular VSWF centered about origin i . Formulas for the OA Mie coefficients and the translation matrix elements appear in the [Appendix](#).

Equation (5), in conjunction with Eqs. (2)-(4), represents the formal solution for the scattered field produced by the sphere ensemble. In the case of equal-sized spheres with equal truncation limits L_S , Eq. (5) forms a system of $2N_S L_S (L_S + 2)$ linear equations for the set of scattering coefficients. The matrix H^{i-j} will be fully populated for an arbitrary translation between j and i , and correspondingly all orders/degrees/modes of the scattered field from a sphere j will (in general) influence a particular order/degree/mode of the field at i . This is in stark contrast to the isolated sphere case, in which each scattering order/degree/mode is excited only by the same harmonic component for the incident field. And it is in this respect that the multiple sphere solution departs – in a practical sense – from the single sphere Mie theory: the latter provides an *explicit* formula for the scattered field, whereas the former gives only an *implicit* relationship. That is, numerical methods (in the form of linear equation solvers) are needed to produce a useable solution. This characteristic has obvious relevance with regard to the order truncation limit L_i . Specifically, closure of Eqs. (5) requires an *a-priori* value of L_i for each sphere. In most situations L_i can be set using a Lorenz-Mie criterion for the isolated sphere i , yet there are specific situations in which interactions among neighboring spheres can have a significant effect on the convergence of Eq. (4) [1]. To accommodate such situations, the code allows for both automatic (based on the Lorenz-Mie criterion) and manual (user-set) specification of L_i .

2.2 Numerical solution of the interaction equations

Iterative methods are used in the code to obtain numerical solutions to Eq. (5). Our experience, and that of others, is that the biconjugate gradient method (BCGM) provides the most reliable, and fastest, solution to Eq. (5), as compared to over/under relaxation and order-of-scattering methods [3]. The number of iterations required for a solution depends on a host of parameters; i.e., the number, size parameters, and refractive indices of the spheres, and the proximity of the spheres to each other. In general, as the spheres become more widely separated the solution will converge faster. An important factor affecting convergence is whether any of the spheres is at or near a resonance mode; such conditions can lead to extremely small convergence rates and may be more effectively solved using direct methods [4].

It is relatively expensive, from a computational operations viewpoint, to calculate the translation matrix elements $H_{mnp'klq}^{i-j}$ for an arbitrary translation from origins j to i . In particular, each element involves a finite summation of a scalar spherical wave function – vector coupling coefficient product (see [Appendix](#)). Because of this, precalculation and memory storage of the translation matrix elements can result in a considerably faster iterative solution to Eq. (5) as compared to the situation in which the elements are calculated on-the-fly during each iteration step. Memory storage of the elements for all sphere pair translations, however, can

become impractical for large scale systems of spheres. It is possible to factor the translation matrices H^{i-j} into rotational and axial translational parts [2], which results in a decoupling of order and degree and leads to both faster matrix–vector multiplication and smaller matrix storage requirements. This factorization is used in the MSTM code. Nevertheless, the memory required to store the complete set of translation matrices will always scale as N_S^2 .

Beginning with version 2.2, the MSTM code employs a hybrid BCGM calculation scheme, designed to minimize memory use without sacrificing execution time, to solve the interaction equations of Eq. (5). The procedure is to first split the interaction with sphere i into near field and far field parts;

$$a_{mnp}^i - \sum_{p'=1}^2 \bar{a}_{npp'}^i \sum_{j \in NF(i)} \sum_{l=1}^{L_j} \sum_{k=-l}^l \sum_{q=1}^2 H_{mnp'klq}^{i-j} a_{klq}^j - \sum_{p'=1}^2 \bar{a}_{npp'}^i \sum_{j \ni NF(i)} \sum_{l=1}^{L_j} \sum_{k=-l}^l \sum_{q=1}^2 H_{mnp'klq}^{i-j} a_{klq}^j = \sum_{p'=1}^2 \bar{a}_{npp'}^i f_{mnp'}^i \quad (6)$$

The criterion to determine whether sphere j is in the near field set $NF(i)$ for sphere i will depend on the translation distance kr_{i-j} and the order truncations L_i and L_j . The translation matrix for the far field set is now approximated by the limiting formula for $kr_{i-j} \gg L_S^2$, which results in the matrix reducing to a dyadic operator, i.e.,

$$H_{mnpklq}(\mathbf{kr}) \approx \mathbf{\Pi}_{mnp}^*(\mu, \phi) \cdot \mathbf{\Pi}_{klq}(\mu, \phi) F(kr) = i^{n-l} (\tau_{mnp}(\mu) \tau_{klq}(\mu) + \tau_{mn3-p}(\mu) \tau_{kl3-q}(\mu)) e^{i(k-m)\phi} \frac{4\pi}{ikr} e^{ikr}, \quad kr \gg (n+l)^2 \quad (7)$$

where $\mu = \cos \theta$ and $\mathbf{\Pi}_{mnp} = (-i)^n (\tau_{mnp} \hat{\theta}, i \tau_{mn3-p} \hat{\phi}) \exp(im\phi)$ is the vector spherical harmonic; definitions appear in the Appendix. The far field form is now used to construct an iterative scheme to Eq. (6), so that

$$a_{mnp}^{i,\nu} - \sum_{p'=1}^2 \bar{a}_{npp'}^i \sum_{j \in NF(i)} \sum_{l=1}^{L_j} \sum_{k=-l}^l \sum_{q=1}^2 H_{mnp'klq}^{i-j} a_{klq}^{j,\nu} - \sum_{p'=1}^2 \bar{a}_{npp'}^i \sum_{j \ni NF(i)} F^{i-j} \mathbf{\Pi}_{mnp'}^{*i-j} \cdot \sum_{l=1}^{L_j} \sum_{k=-l}^l \sum_{q=1}^2 \mathbf{\Pi}_{klq}^{i-j} a_{klq}^{j,\nu} = \sum_{p'=1}^2 \bar{a}_{npp'}^i \left(f_{mnp'}^i + b_{mnp'}^{i,\nu-1} \right) \quad (8)$$

in which superscript ν denotes the ν^{th} -corrected approximation to the solution, and the correction coefficients b_{mnp}^i are given by

$$b_{mnp}^{i,\nu} = \sum_{j \ni NF(i)} \sum_{l=1}^{L_j} \sum_{k=-l}^l \sum_{q=1}^2 \left(H_{mnp'klq}^{i-j} - F^{i-j} \mathbf{\Pi}_{mnp'}^{*i-j} \cdot \mathbf{\Pi}_{klq}^{i-j} \right) a_{klq}^{j,\nu} \quad (9)$$

The general solution strategy starts with $b_{mnp}^{i,0} = 0$, and the BCGM is applied to Eq. (8) to seek a solution for $a_{mnp}^{i,1}$. At convergence of the BCGM for a set error criterion, or after a set number of iterations of the BCGM (whichever comes first), the correction factors $b_{mnp}^{i,1}$ are calculated from Eq. (9), using the current values of $a_{mnp}^{i,1}$, and the BCGM continues with $\nu = 2$. The procedure stop when the both the error in Eq. (8) and the norm of $b^\nu - b^{\nu-1}$ have decreased below a set criterion.

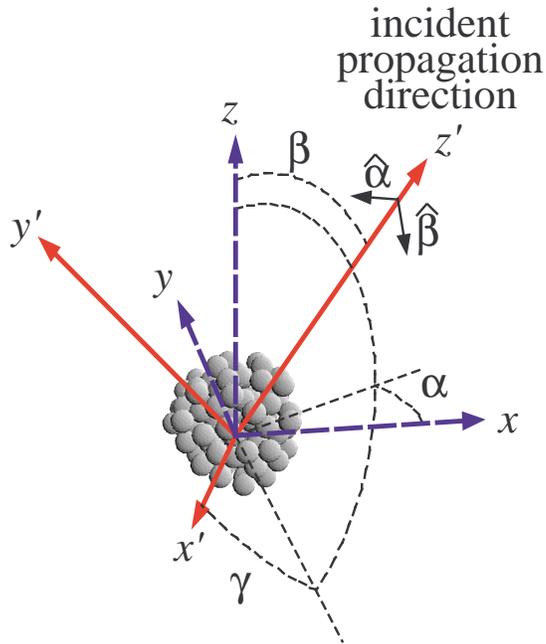


Figure 2: Target and incident field frames

This strategy has two significant advantages in the context of large-scale systems of spheres. First, it significantly reduces the memory requirements, in that it is only necessary to store the near-field translation matrices. Precisely how much memory is required depends on both N_S and the criteria for choosing the near field sets; the latter is controlled via code input options. The vector spherical harmonic functions Π can be efficiently calculated on the fly, and thus little if any advantage is gained by their precalculation and memory storage. Secondly, the interactions involving the far-field translation operators can be performed significantly faster than those involving the exact operators – the former is a dyadic (vector–vector) product, the latter a matrix–vector product – and an iteration of Eq. (8) can be performed in a fraction of the time required for an iteration to Eq. (5). Even though calculation of the correction coefficients in Eq. (9) is expensive – since the exact translation elements are calculated explicitly for each correction stage – the hybrid method typically provides solutions in execution times that are comparable to (and often less than) those obtained from straight BCGM solution of Eq. (5) with stored translation matrices.

2.3 Incident and total scattered field

Referring to Fig. 2, the propagation direction \hat{z}' of the incident field is defined by an azimuth angle α and polar angle β relative to the target coordinate frame. The angle γ appearing in Fig. 2 is used to define the scattering plane, upon which the amplitude and scattering matrix elements are based. The procedure for calculating the amplitude and scattering matrix elements will be discussed in the following section, yet for

now it is noted that determination of these properties, for a set incident direction, requires the solution to Eq. (5) for two mutually orthogonal linear polarizations of the incident field. In the code, the two states correspond to polarization in the $\hat{\beta}$ and $\hat{\alpha}$ directions as illustrated in Fig. 2.

In addition to the standard case of plane wave incidence, the code also provides for the representation of collimated incident beams having a Gaussian amplitude distribution. In the Gaussian beam (GB) case, the incident field expansion coefficients centered about sphere i , appearing in Eqs. (3) and (5), cannot be related to those relative to some other origin by a simple phase shift relation. The general approach used in the code is to define the incident field expansion, for either the plane wave or GB case, relative to the target origin, and then use the VSWF translation theorem to obtain the values of f_{mnp}^i .

At the target origin, the incident field expansion will appear as

$$\mathbf{E}_{inc}(\mathbf{r}) = \sum_{n=1}^L \sum_{m=-n}^n \sum_{p=1}^2 f_{mnp} \mathbf{N}_{mnp}^{(1)}(\mathbf{r}) \quad (10)$$

The order truncation limit L in Eq. (10) – which is chosen so that the expansion will yield an acceptable description of the incident field on each sphere in the ensemble – will typically depend on the size parameter ka_C where a_C is the circumscribing sphere radius illustrated in Fig. 1. For a plane wave (PW), the coefficients for the incident field VSWF expansion, centered about the target origin, are given by

$$\begin{pmatrix} f_{mnp, \hat{\beta}, PW} \\ f_{mnp, \hat{\alpha}, PW} \end{pmatrix} = -4\pi i^{n+1} e^{-im\alpha} \begin{pmatrix} \tau_{mnp}(\cos\beta) \\ i\tau_{mn(3-p)}(\cos\beta) \end{pmatrix} \quad (11)$$

where τ_{mnp} are derived from the vector spherical harmonic functions, and are given by

$$\tau_{mnp}(\cos\beta) = -\frac{1}{4} \left(\frac{2n+1}{\pi} \right)^{1/2} \left((-1)^p \mathcal{D}_{-1m}^{(n)}(\cos\beta) + \mathcal{D}_{1m}^{(n)}(\cos\beta) \right) \quad (12)$$

with $\mathcal{D}_{km}^{(n)}$ denoting the Wigner d -function [5], the definition of which appears in the Appendix.

Along with a propagation direction and polarization angle, the GB is characterized by a focal point (taken here to be the target origin) and beam width ω_0 . For a beam propagating in the $+z$ direction and polarized in the x direction, the amplitude distribution along the focal plane will be given by

$$\mathbf{E}_{inc}(x, y, 0) = \hat{\mathbf{x}} \exp\left(-\frac{x^2 + y^2}{\omega_0^2}\right) \quad (13)$$

The localized approximation is used in the code to provide a VSWF representation of the GB, which is valid for $k\omega_0 \geq 5$ [6, 7]. The incident field expansion coefficients, for the expansion centered about the beam focal point, are given by

$$f_{mnp, \hat{s}, GB} = \bar{g}_n f_{mnp, \hat{s}, PW} \quad (14)$$

$$\bar{g}_n = \exp\left[-\left(\frac{n+1/2}{k\omega_0}\right)^2\right] \quad (15)$$

where \hat{s} denotes the specific polarization state.

Since Eq. (10) is assumed to provide a sufficiently accurate representation of the incident field at all spheres in the target, the sphere-centered expansion for the incident field can be obtained by application of

the translation theorem to [Eq. \(10\)](#). This results in

$$f_{mnp,\hat{s}}^i = \sum_{l=1}^L \sum_{k=-l}^l \sum_{q=1}^2 J_{mnp\,klq}^{i-0} f_{klq,\hat{s}} \quad (16)$$

where J^{i-0} is the regular VSWF translation matrix and f refers to either the PW or GB case.

In a manner analogous to that relating [Eq. \(3\)](#) to [Eq. \(10\)](#), the scattered field from the cluster can be represented by a single outgoing VSWF expansion centered about the cluster origin, so that

$$\mathbf{E}_{sca}(\mathbf{r}) = \sum_{n=1}^{L_T} \sum_{m=-n}^n \sum_{p=1}^2 a_{mnp} \mathbf{N}_{mnp}^{(3)}(\mathbf{r}) \quad (17)$$

$$a_{mnp} = \sum_{i=1}^{N_S} \sum_{l=1}^{L_i} \sum_{k=-l}^l \sum_{q=1}^2 J_{mnp\,klq}^{0-i} a_{klq}^i \quad (18)$$

The truncation limit L_T in the expansion will depend on the distance $|\mathbf{r}|$, with $L_T \rightarrow \infty$ (i.e., lack of convergence) as $|\mathbf{r}| \rightarrow \text{Max } |\mathbf{r}_i|$. In particular, the expansion in [Eq. \(17\)](#) will not be useful to characterize the near-field characteristics of the scattered electric field. However, [Eq. \(17\)](#) is completely valid in the far-field regions, and for this limit L_T becomes equal to the incident field truncation limit L .

2.4 Coordinate rotation, amplitude and scattering matrix, and cross sections

Referring again to [Fig. 2](#), the scattering plane is defined as the $z' - x'$ plane in the incident field coordinate frame. The incident field coordinate frame (x', y', z') , in turn, is obtained by a solid rotation of the target frame (x, y, z) through the Euler angles (α, β, γ) . The expansion coefficients that describe the total scattered field, for incident polarization parallel or perpendicular to the scattering plane, are obtained by

$$a'_{mnp,\parallel} = a'_{mnp,\hat{\beta}} \cos \gamma + a'_{mnp,\hat{\alpha}} \sin \gamma \quad (19)$$

$$a'_{mnp,\perp} = a'_{mnp,\hat{\beta}} \sin \gamma - a'_{mnp,\hat{\alpha}} \cos \gamma \quad (20)$$

in which $\hat{\beta}$ and $\hat{\alpha}$ denote solutions corresponding to the two incident polarization states, and the prime denotes that the coefficients have been rotated from the target to the incident field coordinate frames, in that

$$a'_{mnp,\hat{s}} = \sum_{k=-n}^n \mathcal{D}_{mk}^{(n)}(\cos \beta) e^{ik\alpha} a_{knp,\hat{s}} \quad (21)$$

where $a_{knp,\hat{s}}$ refer to the coefficients obtained from [Eq. \(18\)](#), using the sphere coefficients for incident polarization state \hat{s} . The amplitude matrix elements are obtained by using the the far-field asymptotic form of

the outgoing VSWF, resulting in

$$S_1 = \sum_{n=1}^L \sum_{m=-n}^n \sum_{p=1}^2 (-i)^n a'_{mnp,\perp} \tau_{mn3-p}(\cos \theta') \quad (22)$$

$$S_2 = \sum_{n=1}^L \sum_{m=-n}^n \sum_{p=1}^2 (-i)^{n+1} a'_{mnp,\parallel} \tau_{mnp}(\cos \theta') \quad (23)$$

$$S_3 = \sum_{n=1}^L \sum_{m=-n}^n \sum_{p=1}^2 (-i)^{n+1} a'_{mnp,\perp} \tau_{mnp}(\cos \theta') \quad (24)$$

$$S_4 = \sum_{n=1}^L \sum_{m=-n}^n \sum_{p=1}^2 (-i)^n a'_{mnp,\parallel} \tau_{mn3-p}(\cos \theta') \quad (25)$$

in which θ' denotes the scattering angle. Elements of the scattering matrix can be obtained directly from those of the amplitude matrix following the formulas presented in Bohren and Huffman [8].

The absorption cross section of sphere i is defined so that $C_{abs,i} I_0$ is the rate at which the sphere absorbs energy from the incident wave of irradiance (at the focal point, for a GB) I_0 . For the OA sphere it is given by

$$C_{abs,i} = \frac{2\pi}{k^2} \text{Re} \sum_{n=1}^{L_i} \sum_{m=-n}^n \sum_{p=1}^2 \sum_{q=1}^2 \bar{b}_{npq}^i a_{mnp}^i a_{mnq}^{i*} \quad (26)$$

in which \bar{b}_{np}^i is a property solely of sphere i and is defined by

$$\bar{b}_{npq}^i = - \left([\bar{\mathbf{a}}_n^i]^{-1} + \mathbf{I} \right) = - \left((-1)^{p+q} \frac{\bar{a}_{n3-p3-q}^i}{\bar{a}_{n11}^i \bar{a}_{n22}^i - \bar{a}_{n12}^i \bar{a}_{n21}^i} + \delta_{p-q} \right) \quad (27)$$

The a_{mnp} coefficients in the above formula would correspond to either the parallel or perpendicular polarization values; the absorption cross section for unpolarized incident radiation would be the average of the two. The absorption cross section of the entire ensemble is obtained from the sum of the individual sphere cross sections;

$$C_{abs} = \sum_{i=1}^{N_s} C_{abs,i} \quad (28)$$

In a similar manner, an extinction cross section of an individual sphere can be defined so that $I_0 C_{ext,i}$ is the rate at which the sphere removes energy from the incident wave. The optical theorem applied to the field scattered from the sphere gives

$$C_{ext,i} = -\frac{2\pi}{k^2} \text{Re} \sum_{n=1}^{L_i} \sum_{m=-n}^n \sum_{p=1}^2 a_{mnp}^i p_{mnp}^{i*} \quad (29)$$

As before, the scattering and incident field coefficients would correspond to the particular polarization state. The total ensemble extinction cross section would also be obtained from the sum of the parts;

$$C_{ext} = \sum_{i=1}^{N_s} C_{ext,i} \quad (30)$$

Unlike the absorption cross section, the extinction cross section for the individual sphere would be difficult – if not impossible – to experimentally measure. The definition of this quantity relies on the superposition model of the scattered field, and although this model serves perfectly well as a means to solve Maxwell’s equations for the ensemble, it is not obvious how the ‘partial’ fields scattered from the individual spheres could be discriminated in an experiment. The sphere extinction is also not bound by the isolated–particle inequality of $C_{ext, i} > C_{abs, i}$; it is entirely possible for this inequality to be reversed or for $C_{ext, i} < 0$ on the individual sphere level. And in this respect a scattering cross section is only meaningful on the target level, and is obtained from energy conservation via

$$C_{sca} = C_{ext} + C_{abs} \quad (31)$$

Finally, the *independent* scattering cross section of the sphere is *defined* as

$$C_{i-sca, i} = \frac{2\pi}{k^2} \sum_{n=1}^{L_i} \sum_{m=-n}^n \sum_{p=1}^2 |a_{mnp}^i|^2 \quad (32)$$

and, when multiplied by I_0 , would represent the power transported from the sphere by the sphere’s scattered field. It is important to understand that the $C_{i-sca, i}$ does not have the same additive properties as Eqs. (28) and (30); the sum of the sphere independent scattering cross sections will not equal C_{sca} . The usefulness of $C_{i-sca, i}$ is that it provides, in combination with the absorption and extinction cross sections, a relative measure of the local incoherent (or multiply–scattered) field intensity in the target of spheres. By multiplying Eq. (5) through by a_{mnp}^{i*} and using the definitions of the cross sections, it can be shown that

$$\begin{aligned} C_{ext, i} - C_{i-sca, i} - C_{abs, i} &= \frac{2\pi}{k^2} \operatorname{Re} \sum_{\substack{j=1 \\ j \neq i}}^{N_s} \sum_{k, l, q} \sum_{m, n, p} a_{mnp}^i H_{mnp, klq}^{i-j} a_{klq}^j \\ &= C_{d-sca, i} \end{aligned} \quad (33)$$

in which $C_{d-sca, i}$ denotes the *dependent* scattering cross section of the sphere, and represents the power flowing to/from the sphere due to the interaction of the sphere’s scattered field with the multiply–scattered field. In this respect, the relative magnitudes of $C_{d-sca, i}$ and $C_{ext, i}$ indicates the relative degree to which the sphere is excited by the multiply–scattered and the incident fields.

2.5 The T matrix relationships

Substitution of Eqs. (16) into Eq. (5) leads to

$$T_{mnp\nu}^i - \sum_{p'=1}^2 \bar{a}_{npp'}^i \sum_{\substack{j=1 \\ j \neq i}}^{N_s} \sum_{\nu'} H_{mnp\nu', \nu'}^{i-j} T_{\nu'\nu}^j = \sum_{p'=1}^2 \bar{a}_{npp'}^i J_{mnp\nu'}^{i-0} \quad (34)$$

In the above and what follows, Greek subscripts are shorthand for the degree–order–mode triplet, i.e., $\nu = (klq)$. The sphere–target T^i matrix is defined so that

$$a_{\mu}^i = \sum_{\nu} T_{\mu\nu}^i f_{\nu} \quad (35)$$

and will typically have more columns than rows; the largest row and column order will be L_i and L , respectively.

Replacing Eq. (35) into Eq. (18) results in

$$a_\mu = \sum_{i=1}^{N_S} \sum_{\nu} \sum_{\nu'} J_{\mu\nu'}^{0-i} T_{\nu'\nu}^i f_\nu = \sum_{\nu} T_{\mu\nu} f_\nu \quad (36)$$

The cluster-centered T matrix treats the ensemble of spheres as a single – albeit nonspherical – particle. The cross sections of the target and the scattering matrix – in either fixed or random orientation – can be obtained analytically from its properties. However, the T matrix cannot predict the fields within the cluster or the cross sections of the individual spheres, because Eq. (17) will be valid only for radii that exceed the largest sphere–target origin distance. The detailed individual–sphere and near–field information – which is inaccessible from Eq. (36) – can, however, be obtained from the original superposition model.

Calculation of the target T matrix in the code is accomplished by iterative solution of Eq. (34) for a succession of $\nu = (klq)$ values. The same hybrid BCGM, of the form in Eq. (8), is used to solve Eq. (34). Upon each solution, the column elements of the T matrix corresponding to ν are obtained via the contraction in Eq. (36). The algorithm is described in detail in [9].

2.6 Random orientation

The random orientation cross sections can be obtained by using the matrix relationships for the scattered and incident field, Eqs. (16) and (35), in Eqs. (26) and (29) and integrating the incident field over all propagation and polarization directions. Because the transformation between PW and GB representations in Eq. (14) is independent of propagation direction, the integration can be performed in a general manner without considering the specific form of the incident field. This results in

$$\langle f_{mnp} (f_{m'n'p'})^* \rangle = \delta_{m-m'} \delta_{n-n'} \delta_{p-p'} \bar{g}_n^2 \quad (37)$$

where $\langle \dots \rangle$ denotes orientation averaging and \bar{g}_n is given by Eq. (15); note that the plane wave case will have $\bar{g}_n \rightarrow 1$. The formulas for the individual sphere random orientation cross sections are then

$$\langle C_{abs,i} \rangle = \frac{2\pi}{k^2} \text{Re} \sum_{n=1}^{L_i} \sum_{m=-n}^n \sum_{p=1}^2 \sum_{p'=1}^2 \sum_{\nu} \bar{b}_{npp'}^i T_{nmp\nu}^i T_{nmp'\nu}^{i*} \bar{g}_\nu^2 \quad (38)$$

$$\langle C_{i-sca,i} \rangle = \frac{2\pi}{k^2} \sum_{\mu} \sum_{\nu} |T_{\mu\nu}^i|^2 \bar{g}_\nu^2 \quad (39)$$

$$\langle C_{ext,i} \rangle = -\frac{2\pi}{k^2} \text{Re} \sum_{\mu} \sum_{\nu} J_{\nu\mu}^{0-i} T_{\mu\nu}^i \bar{g}_\nu^2 \quad (40)$$

As before, the total orientation-averaged absorption and extinction cross sections for the cluster will be the sum of the individual sphere values, and the total scattering cross section will be the difference between the total extinction and absorption cross sections.

The random-orientation scattering matrix can be obtained analytically from operations on the T matrix, and is represented as an expansion of generalized spherical functions [9]. The formulas for the expansion coefficients were originally derived for plane wave excitation, yet for GB excitation the generalized spherical function expansion for the scattering matrix can be calculated by making the simple transformation

$$T'_{nmp\,klq} = T_{nmp\,klq} \bar{g}_l \quad (41)$$

and then applying the plane wave formulas to T' .

3 Multiple sphere T -matrix (MSTM) code

3.1 Structure and compilation

The code is organized into the following five components:

`mstm-modules.f90`: Contains modules for data input, special function calculation, iterative linear equation solving, and scattering property calculation.

`mstm-main.f90`: The prepackaged main program, which reads input parameters from an input file, calls the subroutines corresponding to the calculation options, and writes output files.

`mpidefs-parallel.f90`: A module which defines the MPI commands appearing in the `mstm-modules.f90` and `mstm-main.f90` code blocks for use on multiprocessor platforms.

`mpidefs-serial.f90`: A module which defines MPI commands for use on single processor (serial) platforms.

`mstm-intrinsics.f90`: Compiler-specific (non-standard Fortran) functions for command-line argument retrieval and system time operations. The users must modify this module to suit their specific compiler.

The actual file names included in the distribution may also include the version identifier, which will appear as `mstm-modules-v2.1.f90`, etc. The version identifier will be omitted in the following discussion.

Compilation of the code using the GNU *g95* on a MS-Windows, single processor machine would involve

```
g95 -o mstm.exe mpidefs-serial.f90 mstm-intrinsics.f90
    mstm-modules.f90 mstm-main.f90
```

This places the executable in the file `mstm.exe`. Compilation using the MPICH2 package for execution on a parallel machine would use

```
mpif90 -I/opt/mpich2-1.2.1p1/include -g -o mstm.out
    mpidefs-parallel.f90 mstm-intrinsics.f90
    mstm-modules.f90 mstm-main.f90
```

and would put the executable in `mstm.out`.

Other compilers follow the same basic plan. It is important to compile the module files in the order they are given. And please remember that the `mstm-intrinsics.f90` must be modified to match the command-line recognition and retrieval intrinsic functions of the compiler. The distribution has the intrinsics set up for *gfortran*.

3.2 Prepackaged main program

The `mstm-main.f90` program included with the distribution is designed to offer the most common calculation options and output formats, and should serve the computational purposes of most users. Modification of the main program for more specialized types of calculations should be straightforward for the programmer with moderate fortran experience. The code employs a highly modular structure, in which the various tasks

involved in the solution are performed in specialized subroutines, and modification of the code to perform a specific task, such as averaging scattering matrix values over several target configurations or calculating near field values along a non-rectangular domain, will typically involve rearranging subroutine calls to produce the desired output. Customized coding is discussed in [Sec. \(5\)](#).

In using the code with the default main program, the properties of the sphere cluster and run variables are passed to the code using an input file. An input file can be designated by a command line argument; on a serial machine, with the executable named `mstm.exe` and an input file named `mstm-01.inp`, the command to execute the code would appear as

```
mstm mstm-01.inp
```

On a parallel machine the execution command will appear in the shell script used to submit the job. I am only familiar with running the code on the Auburn University College of Engineering HPCC [10], and you will need to consult with the appropriate gurus to set up a parallel job on your cluster.

The input file must be in the same directory as the executable. The default input file is `mstm.inp`; this file must be present if no command line argument is given.

The input file consists of paired lines; the first line of a pair representing a parameter ID, and the second representing the value or option for that parameter. The order of the paired lines is not important. If a pair corresponding to a particular parameter is not present, the code will use the default value.

An example of an input file, showing the first few input parameters, is shown below.

```
number_spheres
100
sphere_position_file
ran100.pos
output_file
test.dat
length_scale_factor
2.d0
real_ref_index_scale_factor
1.6d0
imag_ref_index_scale_factor
0.01d0
mie_epsilon
1.d-3
```

Note that the parameter ID, i.e., `number_spheres` or `output_file`, must appear as written above. A description of the parameters follows; default values are given in parentheses.

3.2.1 Options related to sphere target specification

`number_spheres`: N_S , the number of spheres in the cluster.

`sphere_position_file`: File name containing the sphere size, position, and (optionally) refractive index data. If the filename is blank, or if it is given the value `at_bottom`, sizes, positions, and refractive indices appear as the last lines in the input file, following a parameter ID of `sphere_sizes_and_positions`. The position file, or the appended position information at the bottom of the input file,

should have N_S lines; if the number of lines is smaller than the input N_S , then N_S will be reduced to match the number of lines. All lines in the file must have either 4, 6, or 8 columns. The first four correspond to the radius and X, Y, Z positions of the i^{th} sphere in the list. Units are arbitrary yet must be consistent for radius and position. The 5th and 6th columns, if present, correspond to the real and imaginary parts of the bulk refractive index of the sphere, and the 7th and 8th columns, if present, correspond to the real and imaginary parts of the chiral factor χ , as defined in Eq. (1). If these columns are not present the refractive index of the spheres is taken to be the same for all spheres and given by the refractive index scaling factor, and the chiral factor for the sphere is given by the chiral factor appearing as an input option (see below). Default is `at_bottom`: sphere sizes and positions appearing at the bottom of the input file.

`length_scale_factor`: Dimensionless length scale factor; the radii and positions obtained from the position file are multiplied by this factor, so that the size parameter of the i^{th} sphere is the scale factor times the radius.

`real_ref_index_scale_factor`: Multiplies the sphere real refractive index value from the position file; if refractive index values are explicitly given in the position file, then set this parameter to 1. If refractive index values do not appear in the position file (i.e., 4 column option), then the scale factor becomes the real refractive index value for all spheres.

`imag_ref_index_scale_factor`: Same idea as the above, except now applied to the imaginary part of the refractive index.

`real_chiral_factor`: Multiplies the sphere real chiral factor from the position file; if chiral factors are explicitly given in the position file, then set this parameter to 1. If chiral factor values do not appear in the position file (i.e., 4 or 6 column option), then this parameter becomes the real chiral factor for all spheres. Non active spheres have this value set to zero (0.d0).

`imag_chiral_factor`: Same idea as the above, except now applied to the imaginary part of the chiral factor. Not active spheres have this value set to zero (0.d0).

3.2.2 Options related to numerical solution

`mie_epsilon`: Convergence criterion for determining the number of orders to include in the Mie expansions for each sphere (10^{-4}). Setting `mie_epsilon` to a negative integer $-L$ forces all sphere expansion to include L orders.

`translation_epsilon`: Convergence criterion for estimating the maximum order of the T matrix for the cluster (10^{-3}).

`solution_epsilon`: Error criterion applied to solution of Eq. (8). When the normalized mean square error of the solution, for a given correction step ν , decreases below this value an updated correction term for $\nu + 1$ is calculated. The solution exits when the relative norm of the difference between subsequent corrections ν and $\nu + 1$ decreases below this value. (10^{-7}).

`iterations_per_correction`: The maximum number of BCGM iterations to Eq. (8) for a given correction stage ν . A recalculation (correction) via Eq. (9) will be performed when the BCGM iteration count reaches this value or when the BCGM error drops below `solution_epsilon`, whichever comes first (20).

max_number_iterations: The maximum *total* number of BCGM iterations to Eq. (8) (summing over all correction stages). The code will send a message if the maximum number of iterations is exceeded (2000).

near_field_distance: Real parameter; the set of spheres $NF(i)$ in the near field of sphere i , for which the exact translation operator is applied in Eq. (8), is determined as all spheres with origin-to-origin distances kr less than or equal to this value. There are several special options for this parameter:

- A value larger than the maximum sphere-to-sphere kr (i.e., a sufficiently large number, say 10^6) will result all spheres belonging to the near field set. In this case, the solution reduces to the BCGM applied directly to Eq. (5), without far-field corrections. The `iterations_per_correction` parameter becomes inapplicable in this case.
- A value smaller than the minimum sphere-to-sphere kr (i.e., zero) will result in all spheres belonging to the far field set. Interactions, during BCGM solution of Eq. (8), will be calculated solely with the far field translation operator. This will typically result in the smallest iteration execution time, yet the convergence rate of the solution will suffer – especially if the spheres are close to each other. In particular, using this setting for clusters of contacting spheres may result in a non-convergence of the BCGM.
- A negative value (i.e., -1.) will apply an automatic criterion to selecting the spheres. The set $NF(i)$ for sphere i is set under this criterion as the spheres whose origin-to-origin distance to i is given by

$$kr_{i-j} \leq \left[\frac{1}{2} (L_i + L_j) \right]^2 \quad (42)$$

The default value is -1. (automatic selection). Those performing time-intensive calculations on large sphere clusters should experiment with different values of this parameter.

max_memory_per_processor: Integer; the maximum memory used for translation matrix storage for each processor, in MB. Relevant only for parallel runs, this quantity should be somewhat less than the total memory available to a single processor (1500). This quantity affects how the processors are grouped for parallel computation of the cluster T matrix, as described in Sec. (3.3).

store_translation_matrix: Integer switch: =0, the near field translation matrices in Eq. (8) are calculated during iteration and not stored in memory; =1, near field matrices are calculated prior to the solution and stored in memory, and far field matrices are calculated on the fly; =2, both near and far field matrices are stored in memory. Use 0 if you run into problems with too much memory being used; this will result in the code running slower (execution time will typically double), yet the no-store option will allow for almost unlimited sphere numbers (1).

3.2.3 Options related to incident field and calculated quantities

min_scattering_angle_deg: The starting value of the scattering angle for scattering matrix computations, in degrees (0.0).

max_scattering_angle_deg: Ending value of scattering angle, in degrees (180.0).

number_scattering_angles: The number of scattering angles, evenly spaced between the minimum and maximum values, for scattering matrix calculation (181).

normalize_scattering_matrix: Integer switch: = 1, scattering matrix values S_{ij} for $(ij) \neq (11)$ are scaled by S_{11} ; = 0, matrix values are unscaled (1).

gaussian_beam_focal_point: $X, Y,$ and Z coordinates of the Gaussian-profile incident beam, relative to the origin and scaling in the sphere position file (i.e., before **length_scale_factor** has been applied). This array defines the origin of the T matrix, and random orientation results are based upon this origin. (0.0,0.0,0.0).

gaussian_beam_constant: Dimensionless parameter $C_B = 1/k\omega_0$ (Eq. (13)) which characterizes the inverse width, at the focal point, of an incident Gaussian profile beam. Setting $C_B = 0$ selects plane wave incidence. The localized approximation used to represent the Gaussian beam is accurate for $C_B \leq 0.2$. Default is the plane wave condition (=0.0), and this number is not scaled by the length scaled factor. Note that Gaussian beam options apply to both fixed orientation and random orientation calculations.

fixed_or_random_orientation: Integer switch: = 0 for a fixed orientation, = 1 for random orientation results via the T matrix scheme. When = 0, input parameters corresponding to the T matrix solution are not pertinent to the run, and likewise for the fixed orientation parameters when = 1 (0).

output_file: File name for file to which final calculation results are written; see Sec. (3.4.2) (**test.dat**).

run_print_file: File name for file to which intermediate output results are written; see Sec. (3.4.1). If blank, results are written to standard output (the screen). Default is blank.

3.2.4 Options for fixed orientation calculations

incident_azimuth_angle_deg: The azimuth angle α of the incident field propagation direction, relative to the sphere cluster coordinate system, in degrees (0.0).

incident_polar_angle_deg: Polar angle β for propagation direction, degrees (0.0).

scattering_plane_angle_deg: Angle γ which sets the scattering plane for scattering matrix calculations, per Fig. 2 and accompanying discussion. Note that if $\alpha = \beta = 0$, γ corresponds to the azimuth angle ϕ of the scattering plane relative to the cluster coordinate system (0.0).

calculate_scattering_coefficients: Integer switch selecting whether the sphere scattering coefficients a_{mp}^i are calculated from solution to Eq. (5) (=1), or read from a file generated from a previous solution (=0). The latter option is useful for generating near field maps on different planes without having to recalculate the scattering coefficients (1).

scattering_coefficient_file: File name for the file to which the scattering coefficients are written (**amn-temp.dat**).

track_iterations: Integer switch selecting whether the error after each iteration to Eq. (5) is displayed in the run file (=1), or suppressed (=0). This option allows the user to track the convergence of the solution (1).

calculate_near_field: Integer switch for calculation of near field: = 0 for no near field calculations, = 1 to select near field calculations. The following 8 input parameters are pertinent only when the near field is calculated (0).

near_field_plane_coord: Near field values are calculated in a rectangular grid lying in the plane denoted by this integer value, = 1: $\hat{y} - \hat{z}$ plane; = 2: $\hat{z} - \hat{x}$ plane; = 3: $\hat{x} - \hat{y}$ plane (1).

near_field_plane_position: The distance of the calculation plane from the cluster coordinate origin, scaled by k and relative to the target coordinate system defined in the sphere position file (0.0).

near_field_plane_vertices: Two pairs of numbers, (kX'_1, kY'_1) , (kX'_2, kY'_2) which denote the vertices (opposite corners) of the rectangular region, in the near field plane, in which field calculations are made. X' and Y' refer to either (y, z) , (z, x) , or (x, y) for **near_field_plane_coord** = 1,2,3. The coordinates in the first pair must be smaller than those in the second pair. The coordinates are not scaled by the length scale factor; they are implicitly in size parameter units (i.e., scaled by k). Again, the positions are relative to the target frame. $(-10.0, -10.0, 10.0, 10.0)$.

spacial_step_size: The spacial step size $k\Delta x$ of calculation grid points (0.1).

polarization_angle_deg: A specific polarization state of the incident field is needed to calculate the near field. The field is taken to be linearly polarized, with a polarization angle of γ relative to the $\hat{k}-\hat{z}$ plane. When $\beta = 0$, γ becomes the azimuth angle of the incident electric field vector relative to the cluster coordinate system (0.0).

near_field_output_file: File name for the output electric field values. The format for the file is described in [Sec. \(3.4.3\)](#) (`nf-temp.dat`).

near_field_output_data: Integer switch specifying the data to be written to the output file, = 0, $|\mathbf{E}|^2$; = 1, complex \mathbf{E} vector; = 2, complex \mathbf{E} and \mathbf{H} vectors. See [Sec. \(3.4.3\)](#) (1).

plane_wave_epsilon: The incident field component for the near field calculations – for either the plane wave or Gaussian beam models – is calculated using a single, regular VWH expansion centered about the beam focal point. The plane wave epsilon is a convergence criterion for this expansion (0.01).

3.2.5 Options for random orientation calculations

calculate_t_matrix: Integer switch selecting whether the T matrix is read from a file (= 0), calculated in its entirety and written to a file (= 1), or calculated beginning with the next largest order of a partially-calculated T matrix read from a file, and appended to the same file (= 2). Option 0 allows for calculation of random properties for different incident beam configurations (plane wave or Gaussian) without having to recalculate the T matrix. Option 1 calculates the T matrix elements using the sequential solution of [Eqs. \(34\)](#) until a set convergence criterion is reached. Option 2 is included for situations in which option 1 is interrupted prior to convergence; the calculations will pick up where the interrupted run left off and continue until convergence (1).

t_matrix_file: File name for the file to which the T matrix is read (option 0), written (option 1), or read and appended (2). Note that the T matrix will be written to a file regardless of whether it is intended to be used again in subsequent runs: the file serves as temporary storage of T matrix columns during calculation. *This file can be very large* – on the order of a few GB – so make sure you have adequate disk space. (`tmatrix-temp.dat`).

t_matrix_convergence_epsilon: Calculation of the T matrix is accomplished by solution of the interaction equations for a sequence of right hand sides, with each RHS corresponding to the order l , degree

k , and mode q component of a generalized plane wave expansion centered about the focal point. For each order l the random-orientation extinction and scattering efficiencies of the cluster are calculated, and a converged T matrix is identified when the absolute difference in the efficiencies, from one order to the next, decreases below this convergence epsilon (10^{-6}).

3.2.6 Termination of input data

Reading of options from the input file will terminate without error when the end-of file is reached. Alternatively, the input can be terminated by using the parameter ID of `end_of_options`; the input process will be closed when this line is reached, and ID/parameter values located after this line will have no influence on the run. This statement is useful for quick modification of an input file, in that ID/parameter pairs can be shuffled either before or after the `end_of_options` line within the same file to set up different runs. The `sphere_sizes_and_positions` ID has the same effect as `end_of_options` when sphere sizes and positions are read from a separate file, yet when the sizes/positions are appended to the input file the ID preceding the data must be `sphere_sizes_and_positions`.

3.3 Parallel considerations

The code employs parallelization during four computational tasks: 1) the matrix-vector product $H^{i-j} a^j$ appearing in Eq. (5); 2) the solution of Eq. (34) for the different right-hand-side vectors; 3) computation of the expansion coefficients for the random orientation scattering matrix representation; and 4) calculation of the near field values. The last two steps involve a straightforward distribution of non-recursive computational tasks among the N_P processors used in the run. The first two tasks, however, occur simultaneously during calculation of the T matrix. The strategy used is to subdivide the N_P processors via $N_P = N_1 \cdot N_2$. Each member of the N_2 group is involved in a solution, for given right-hand-side, to Eq. (34), and associated with this member are the N_1 processors which are used to perform the matrix-vector product during iteration. The maximum efficiency is obtained when N_1 is made as small as possible and, by extension, N_2 as large as possible; this minimizes the overall amount of data transfer among processors required to complete a T matrix calculation. When the `store_translation_matrix = 1` option is used, the minimum value of N_1 will be determined by the ratio of the memory required to store the complete set of translation matrix elements to the memory available to a single processor, with the latter quantity user-set by the variable `max_memory_per_processor`. On the other hand, the option `store_translation_matrix = 0` automatically results in $N_1 = 1$ and $N_2 = N_P$.

In fixed orientation calculations the interaction equations are solved for two mutually orthogonal incident polarization states. Hence, the code attempts to set the group number $N_2 = 2$ and $N_1 = N_P/2$; this is done automatically if the translation matrices are not stored in memory. Also note that if $N_1 > N_S$, then $N_1 - N_S$ processors will be idle during solution of Eq. (5). All processors will be put to use for subsequent near field calculations, if performed.

3.4 Output

3.4.1 Run (intermediate) print file

The purpose of the run output is to inform the user of the options and parameters of the calculation, including the number of equations in Eq. (5), the memory used to store the translation matrices, and an estimate of the time required to complete the calculation. Intermediate output, of a form dependent on the

run options (i.e., fixed or random) will appear in the run print file during execution. Warnings and errors will also appear here.

Runs on parallel platforms should have the run print file set to a filename as opposed to the standard output; parallel jobs typically have the standard output redirected to a file, and choosing a set run print file will result in a somewhat cleaner output than what would be produced by redirection. Each write statement to the run file is followed, in the code, by a flush statement, and this allows the user to follow the output, without appreciable buffer delay, by refreshing the file in the viewing application.

3.4.2 Output file

Information included in the output file includes

- The relevant run parameters and options for the run, i.e., fixed or random, incident angles, GB, etc.
- A listing of the sphere properties, enumerated in the same order as given in the position file. Sphere data includes the size parameter and k – scaled position with respect to the target origin (i.e., as defined in the input file), L and R refractive indices (only the bulk value will appear for systems that are completely non active), the unpolarized extinction, independent scattering, and absorption efficiencies, and the ratio of actual to Lorenz-Mie absorption efficiencies. For fixed orientation the efficiencies are computed from the average of the β and α incident polarization values, and will correspond to values for unpolarized incident radiation. Both the fixed and random orientation efficiency results account for the specified GB conditions.
- The total extinction, scattering, and absorption efficiencies of the target, for unpolarized incident radiation and defined with respect to the volume–mean size parameter, and the asymmetry parameter.
- For fixed orientation calculations, the total efficiency factors for incident polarization parallel and perpendicular to the scattering plane (new in revision 1.2).
- The scattering matrix values for the set range of scattering angles. The phase function S_{11} is normalized so that

$$\frac{1}{2} \int_0^\pi S_{11}(\theta) \sin \theta d\theta = 1 \quad (43)$$

When `normalize_scattering_matrix = 1`, the remaining elements are scaled by S_{11} at the same angle, i.e., $S_{12}(\theta)/S_{11}(\theta)$. The listing is limited to the matrix values S_{ij} with $j \geq i$. In the case of random orientation the lower diagonal elements can be obtained from symmetry. Users who need the full scattering matrix for fixed orientation will need to modify the output statements in the main program.

- For random orientation runs, the coefficients for the generalized spherical function expansion of the random orientation scattering matrix. The expansion includes orders 0 to $2L$, where L is the order of the T matrix. Users can consult the main program to see the use of these coefficients in calculating scattering matrix values.

3.4.3 Near field file

The first line in the near field file contains the pair $N_{F,X'}$ and $N_{F,Y'}$, which are the number of data points in the X' and Y' directions:

$$N_{F,X'} = 1 + \frac{X'_2 - X'_1}{\Delta x}, \quad N_{F,Y'} = 1 + \frac{Y'_2 - Y'_1}{\Delta x} \quad (44)$$

The second line contains the number $N_{I.S.}$, which is the number of spheres with boundaries that intersect the near field plotting plane. The next $N_{I.S.}$ lines contain $kX'_{i,I.S.}$, $kY'_{i,I.S.}$, and $R_{i,I.S.}$, which denote the X' and Y' position of the i^{th} intersecting sphere's origin when projected onto the plotting plane, and the radius of the circle formed on the plane by the intersection of the sphere. This information can be used with a graphics package to draw the locations of the sphere boundaries on top of the field plots. The remaining $N_F = N_{F,X'} \cdot N_{F,Y'}$ lines in the file contain the calculation points and field values. The first pair of numbers is the position kX' , kY' of the data point, and these numbers are relative to the target frame as defined by the sphere position file. The remaining columns for the line depend on the choice of `near_field_output_data`. For option 0 the single number $|\mathbf{E}|^2$ is written, option 1 writes the real and complex parts of the 3 cartesian components of the complex vector \mathbf{E} (six numbers in all), and option 2 writes the complex electric \mathbf{E} and magnetic \mathbf{H} field vectors (12 numbers). The cartesian components are based on the target reference frame. The electric and magnetic field values are scaled to the corresponding field amplitudes of the incident field at the target origin. Using this convention, the magnetic field values for a general VSWF expansion are calculated by

$$\mathbf{H}(\mathbf{r}) = \frac{m}{i} \sum_{n=1}^m \sum_{m=-n}^n \sum_{p=1}^2 g_{mnp} \mathbf{N}_{mn3-p}(\mathbf{r}) \quad (45)$$

in which g denote the coefficients of the expansion (a , f , or the internal field coefficients for regions inside a sphere) and m is the local refractive index, which will be 1 for regions outside a sphere. Observe that the operation $3 - p$ switches the mode from one to the other. Calculation of the internal fields (electric and magnetic) for optically active media is somewhat more complicated than that implied by the above equation; refer to the [Appendix](#) for details.

4 Examples

Please note that the calculation examples provided in this section may have been (and were probably) performed using previous versions of the MSTM code. Because of this, the specific appearance of the input and output information may not coincide with that obtained from your version of the code.

4.1 Calculation of random orientation properties for a spherical cluster

The first example will demonstrate how MSTM is used to calculate the random-orientation properties of a spherical cluster of spheres. The calculations were run using the original version of the MSTM code, and the run and output files will show some differences with the current version. The target is shown in [Fig. 3](#), and consists of $N_S = 375$ non-active spheres distributed uniformly into a spherical volume with an average volume fraction of $c = 0.1$. The spheres are identical, with size parameter $x_S = 2$ and refractive index $m = 1.31 + 0.1i$. The size parameter of the circumscribing sphere of the target is $x_C = x_S(N_S/c)^{1/3} = 31.1$.

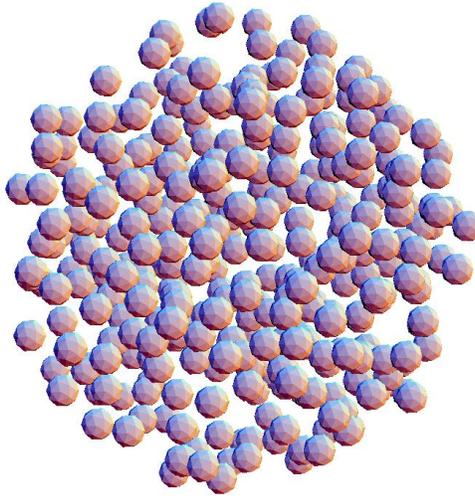


Figure 3: $N_S = 375$ spherical cluster

The input file for the run is

```
number_spheres
375
sphere_position_file
ran375fvp1.pos
output_file
example_1.dat
run_print_file
run.dat
length_scale_factor
2.d0
real_ref_index_scale_factor
1.31d0
imag_ref_index_scale_factor
0.1d0
fixed_or_random_orientation
1
min_scattering_angle_deg
0.d0
max_scattering_angle_deg
180.d0
number_scattering_angles
361
gaussian_beam_constant
0.0d0
gaussian_beam_focal_point
0.d0,0.d0,0.d0
```

```
calculate_t_matrix
1
```

Options/parameters not listed are taken to be the default values; the options relating to the GB are actually the default conditions (i.e., a plane wave) and could have been omitted. Options relating to fixed orientation calculations – which are not relevant to this run – could also have appeared in the file with no harm done.

The run was executed on the COE HPCC on 128 processors. The file `run.dat` provides a listing of the intermediate results:

```
input file is mstm.inp
position data has 4 records
number of spheres, volume size parameter: 375 0.14422E+02
position file:ran375fvpl.pos
output file:example_1.dat
length, ref. indx. scale factors: 2.000 1.310 0.100
thetamin, thetamax, num. theta: 0.0 180.0 361
epsmie, epssoln, max number iterations: 0.1000E-03 0.1000E-09 2000
plane wave incidence
random orientation calculations
t matrix calculated, stored in file tmatrix-temp.dat
t matrix convergence epsilon: 0.1000E-03
number of processors, number groups, memory per processor: 128 128 917.796
maximum translation matrix storage: 917.7960 MB
minimum translation matrix storage: 917.7960 MB
maximum sphere order: 5
estimated T matrix order: 39
number of equations: 26250
time per iteration: 5.45 sec
time per group solution: 1.65 min
estimated t matrix calculation time: 41.28 min
```

n	# its	qxct	qabs	qsca	error	est. time rem.
1	16	0.34450E-01	0.11447E-01	0.23003E-01	0.34450E-01	41.28 min
2	16	0.87042E-01	0.29132E-01	0.57910E-01	0.52592E-01	41.28 min
3	16	0.15827E+00	0.53259E-01	0.10501E+00	0.71231E-01	41.28 min
4	16	0.24947E+00	0.84319E-01	0.16515E+00	0.91197E-01	41.28 min
5	16	0.36046E+00	0.12224E+00	0.23822E+00	0.11099E+00	41.28 min
6	16	0.49013E+00	0.16672E+00	0.32340E+00	0.12967E+00	41.28 min
7	16	0.63884E+00	0.21770E+00	0.42114E+00	0.14871E+00	41.28 min
8	16	0.80638E+00	0.27489E+00	0.53150E+00	0.16755E+00	39.52 min
9	16	0.99261E+00	0.33865E+00	0.65396E+00	0.18623E+00	39.52 min
10	16	0.11980E+01	0.40888E+00	0.78915E+00	0.20542E+00	39.52 min
11	14	0.14224E+01	0.48543E+00	0.93692E+00	0.22432E+00	40.83 min
12	14	0.16638E+01	0.56818E+00	0.10956E+01	0.24146E+00	40.83 min
13	15	0.19214E+01	0.65672E+00	0.12646E+01	0.25754E+00	36.57 min
14	15	0.21965E+01	0.75106E+00	0.14454E+01	0.27516E+00	36.57 min
15	15	0.24885E+01	0.85160E+00	0.16369E+01	0.29199E+00	36.57 min
16	15	0.27937E+01	0.95747E+00	0.18362E+01	0.30516E+00	36.88 min

(and after some time...)

```
35 14 0.74080E+01 0.27411E+01 0.46668E+01 0.15455E-02 10.28 min
```

```

36 15 0.74084E+01 0.27415E+01 0.46670E+01 0.48490E-03 9.07 min
37 15 0.74086E+01 0.27416E+01 0.46670E+01 0.14070E-03 6.65 min
38 15 0.74086E+01 0.27416E+01 0.46670E+01 0.37832E-04 4.51 min
T matrix converged, order: 38
execution time: 46.72 min
t matrix read time: 20.85 sec
d matrix calculation, order+degree per proc.: 46.32
d matrix time: 10.26 sec
scat matrix coef time: 1.88 sec

```

The processors for this particular run are set up so that each processor independently performs a solution to Eq. (34), i.e., $N_1 = 1$ and $N_2 = 128$ (Sec. (3.3)). This requires the processor to store the complete set of translation matrices, which amount to around 1 GB. The number of equations in the listing refers to the number of unknowns in Eqs. (5). The columns show the current order to the T matrix solution, the average number of iterations required for a solution, the current values of random-orientation efficiency factors (defined relative to the volume mean size parameter), the error, and the estimated time required for the solution to complete. The time estimate is just that – it can be too low or too high – but it typically is in the right ballpark. Note that the first set of solutions to Eqs. (34) performed by the 128 processors covered orders 1-7 (7 orders would require $2 \cdot 7 \cdot 9 = 126$ solutions, including degree and mode), and the first seven orders listed therefore have the same estimated time remaining. The T matrix for this run converged in 38 orders, and the actual execution time was 46.7 min. The last few lines list the times associated with calculating the expansion coefficients for the random orientation scattering matrix; this calculation will typically be a small fraction of the overall computation.

The output file is `example_1.dat`, and a selection of the file is shown in Fig. 4. The sphere positions for the position file used in the calculations are ordered from the center of the cluster outwards, and the results show a relative decrease in absorption for spheres closer to the center; this is expected because propagation of radiation into the cluster is attenuated by absorption.

4.2 Electric field distribution in a slab of spheres

This example run calculates the electric field distribution in and about a cylindrically-shaped cluster. Again, the run was performed using the original MSTM code, and expect some differences between the output formal shown here and what you get with the newer versions. The target, illustrated in Fig. 5, consists of $N_S = 3000$ identical spheres with $x_S = 4$, $\mathbf{m} = 1.6 + 0.01i$, and $\chi = 0$, uniformly and randomly packed into a circular cylinder of radius = axial length, with an average volume fraction of 0.5. The target is excited with an $\hat{\mathbf{x}}$ -polarized Gaussian profile beam of width $k\omega_0 = 20$ (`gaussian_beam_constant=0.05`) which propagates along the axis of the cylindrical target and is focussed on the target center.

The input file for the run is as follows

```

number_spheres
3000
sphere_position_file
cyl3000fvp5.pos
output_file
example_2a.dat
run_print_file
run.dat
length_scale_factor

```

```

ran375fvp1.pos
output file:
example_1.dat
length, ref. indx. scale factors:
2.000 1.310 0.100
thetamin, thetamax, num. theta:
0.0 180.0 361
epsmie, epssoin, max number iterations:
0.1000E-03 0.1000E-09 2000
plane wave incidence
random orientation calculations
t matrix calculated, stored in file
tmatrix-temp.dat
t matrix convergence epsilon:
0.1000E-03
sphere S.P., pos. (x,y,z), ref. index, Qext, Qsca, Qabs, Qabs/LM
1 2.0000 0.0000 0.0000 0.0000 1.3100 0.1000 0.10087E+01 0.26675E+00 0.29631E+00 0.5226
2 2.0000 -0.0410 -0.0254 -4.4516 1.3100 0.1000 0.10135E+01 0.27136E+00 0.30091E+00 0.5307
3 2.0000 -4.2862 -1.7329 1.2910 1.3100 0.1000 0.10196E+01 0.28554E+00 0.31585E+00 0.5571
4 2.0000 1.5268 4.5862 3.1316 1.3100 0.1000 0.10430E+01 0.28520E+00 0.31553E+00 0.5565

(more lines of single sphere data)

372 2.0000 16.7016 21.8880 -14.0950 1.3100 0.1000 0.10057E+01 0.41317E+00 0.45802E+00 0.8078
373 2.0000 2.9126 30.5920 -3.8676 1.3100 0.1000 0.10105E+01 0.41457E+00 0.45963E+00 0.8107
374 2.0000 -2.0096 29.2300 10.0800 1.3100 0.1000 0.10268E+01 0.39928E+00 0.44265E+00 0.7807
375 2.0000 5.9862 16.6838 -25.4620 1.3100 0.1000 0.10139E+01 0.42103E+00 0.46751E+00 0.8246

total ext, abs, scat efficiencies, w.r.t. xv, and asym. parm
0.74086E+01 0.27416E+01 0.46670E+01 0.90907E+00
scattering matrix elements
theta s11 s22 s33 s44 s21 s32 s31 s42 s41
0.00 0.5955E+02 0.1000E+01 0.1000E+01 0.1000E+01 -0.1151E-04 0.0000E+00 0.0000E+00 0.0000E+00 0.6108E-04
0.50 0.5859E+02 0.1000E+01 0.1000E+01 0.1000E+01 -0.3356E-04 -0.1169E-04 0.3421E-04 0.1241E-08 0.1798E-10 0.6107E-04
1.00 0.5577E+02 0.1000E+01 0.1000E+01 0.1000E+01 -0.1343E-03 -0.1222E-04 0.1377E-03 0.5026E-08 0.4036E-10 0.6101E-04
1.50 0.5134E+02 0.1000E+01 0.1000E+01 0.1000E+01 -0.3027E-03 -0.1312E-04 0.3132E-03 0.1155E-07 -0.3886E-10 0.6091E-04
2.00 0.4564E+02 0.1000E+01 0.1000E+01 0.1000E+01 -0.5391E-03 -0.1445E-04 0.5657E-03 0.2118E-07 -0.4380E-09 0.6075E-04

(more lines of scattering matrix data)

178.50 0.1665E-02 0.6225E+00 -0.6223E+00 -0.2470E+00 0.9107E-02 0.1497E-05 -0.6214E-02 0.4856E-05 -0.7344E-04 -0.2340E-02
179.00 0.1689E-02 0.6194E+00 -0.6194E+00 -0.2398E+00 0.4139E-02 0.3002E-06 -0.2816E-02 0.2644E-05 -0.3256E-04 -0.2327E-02
179.50 0.1704E-02 0.6176E+00 -0.6176E+00 -0.2354E+00 0.1049E-02 0.1893E-07 -0.7122E-03 0.7962E-06 -0.8127E-05 -0.2319E-02
180.00 0.1709E-02 0.6170E+00 -0.6170E+00 -0.2339E+00 0.1044E-12 0.1866E-27 -0.7087E-13 0.7651E-16 -0.8052E-15 -0.2316E-02
scattering matrix expansion coefficients
w a11 a22 a23 a33 a32 a44 a12 a13 a24 a14
0 0.4667E+01 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.4534E+01 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.1248E-03
1 0.1273E+02 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.1268E+02 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.2969E-03
2 0.1933E+02 0.7498E-01 0.2142E+02 -0.2034E-08 -0.4474E-03 0.1944E+02 -0.6211E+00 0.2837E+00 -0.5535E-04 0.1097E-04 0.8525E-03
3 0.2574E+02 0.1080E+00 0.2647E+02 -0.1340E-08 -0.2164E-03 0.2562E+02 -0.5417E+00 0.1565E+00 0.1092E-03 0.9939E-04 0.2072E-02

(the rest of the scattering matrix expansion coefficients, up to w = 2L = 76)

```

Figure 4: Output file listing for example 1.

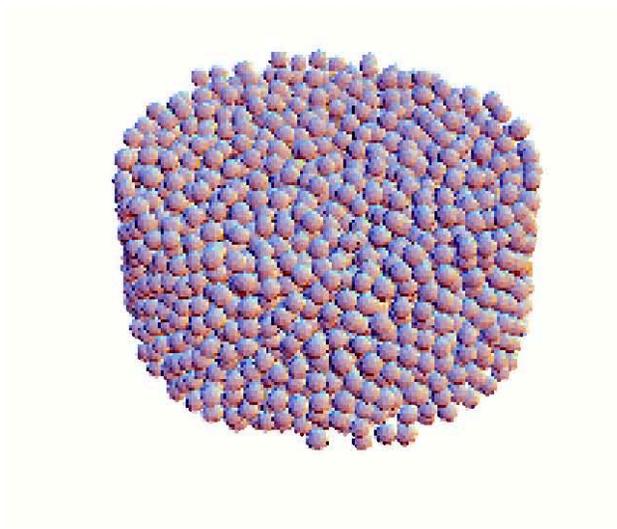


Figure 5: $N_S = 3000$ sphere target

```
4.d0
real_ref_index_scale_factor
1.6d0
imag_ref_index_scale_factor
0.01d0
fixed_or_random_orientation
0
track_iterations
1
calculate_scattering_coefficients
1
scattering_coefficient_file
amn_cylslab.dat
scattering_plane_angle_deg
0.d0
incident_azimuth_angle_deg
0.d0
incident_polar_angle_deg
0.d0
gaussian_beam_constant
0.05d0
gaussian_beam_focal_point
0.d0,0.d0,0.d0
calculate_near_field
1
near_field_plane_coord
1
```

```
near_field_plane_position
0.d0
near_field_plane_vertices
-80.d0,-80.d0,80.d0,80.d0
spacial_step_size
.2d0
polarization_angle_deg
0.d0
near_field_output_file
nf-temp.dat
near_field_output_data
1
plane_wave_epsilon
0.01d0
```

The job was run on 128 processors. After about 10 minutes of run time, the intermediate output file `run.dat` contains

```
input file is mstm.inp
position data has 4 records
number of spheres, volume size parameter: 3000 0.57690E+02
position file:cyl3000fvp5.pos
output file:example_2a.dat
length, ref. indx. scale factors: 4.000 1.600 0.010
thetamin, thetamax, num. theta: 0.0 180.0 181
epsmie, epssoln, max number iterations: 0.1000E-03 0.1000E-09 2000
gaussian incident beam: 1/width: 0.0500
beam focal point: 0.000 0.000 0.000
fixed orientation calculations
scattering plane, incident alpha, beta: 0.00 0.00 0.00
common expansion epsilon: 0.1000E-02
scattering coefficients calculated, stored in file amn_cylslab.dat
near field calculated, stored in file nf-temp.dat
near field data output option: 2
near field plane, position: 1 0.000
near field plane vertices: -80.000 -80.000 80.000 80.000
spacial step size: 0.2000
polarization angle, deg.: 0.00
plane wave epsilon: 0.1000E-01
number of processors, number groups, memory per processor: 128 1 1107.756
maximum translation matrix storage:1134.3418 MB
minimum translation matrix storage:1087.0775 MB
maximum sphere order: 7
estimated T matrix order: 101
number of equations: 378000
time per iteration: 38.04 sec
iter, err: 1 0.30257E+02
iter, err: 2 0.25037E+03
iter, err: 3 0.35267E+04
iter, err: 4 0.58164E+01
iter, err: 5 0.74634E+01
```

```
iter, err: 6 0.37805E+02
iter, err: 7 0.14465E+03
iter, err: 8 0.27319E+02
```

This shows that the calculation consists of solving 3.78×10^5 complex-valued equations for the two incident polarization states. This solution will take some time to converge via iteration – almost nine hours, in this case – and that is why the `track_iterations` option is set; it allows one to gauge whether the solution is getting anywhere. The final lines of the run file appear as

```
iter, err: 850 0.37878E-09
iter, err: 851 0.20156E-09
iter, err: 852 0.17621E-09
max iterations, soln error: 937 0.84522E-10
execution time: 8.576 hours
near field calculations
plane, position: 1 0.000
rectangular plot vertices:
min: -80.000 -80.000
max: 80.000 80.000
number of plotting points, step size: 641601 0.200
max plane wave order: 127
estimated time remaining: 6.79 min
estimated time remaining: 4.96 min
estimated time remaining: 3.17 min
estimated time remaining: 1.56 min
estimated time remaining: 0.00 sec
```

The near field grid specified in the input file consists of the square on the $y - z$ plane of $-80 \leq ky \leq 80$ and $-80 \leq kz \leq 80$, with a step size of $\Delta ky = \Delta kz = 0.2$. The polarization angle is 0, i.e., in the $\hat{\beta}$ direction, and since $\beta = 0$ (incidence along the z axis), the incident polarization points in the x direction. The plane on which the field is mapped is therefore perpendicular to the incident polarization direction. Figure 6 shows results from the calculations. Plotted is the real part of the x component of electric field. The plot on the left is the grid generated by the original input file, and the plot on the right provides a higher resolution map in the region of the focal point. Numbers for the second plot were generated from a recalculation of the field distribution using the scattering coefficients generated from the original calculation. Per the information in `run.dat`, calculation of the near field values required around 7 minutes, as opposed to around 9 hours to calculate the scattering coefficients. The input file for this second run is

```
number_spheres
3000
sphere_position_file
cyl3000fvp5.pos
output_file
temp.dat
run_print_file
run.dat
length_scale_factor
4.d0
real_ref_index_scale_factor
1.6d0
imag_ref_index_scale_factor
```

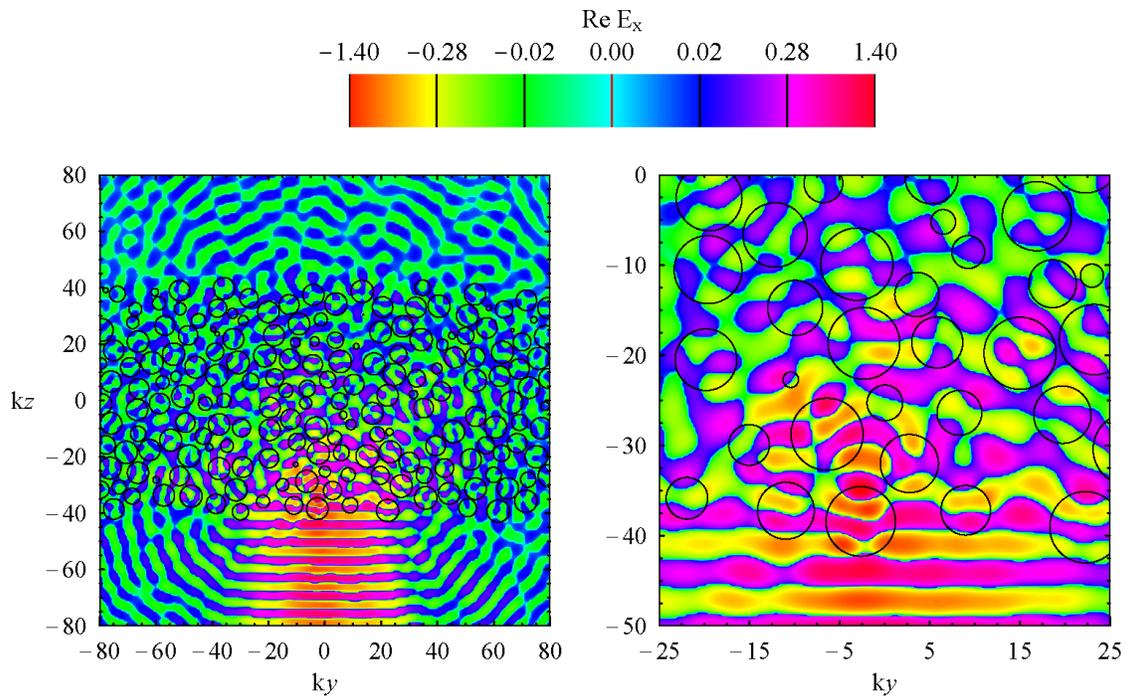


Figure 6: $\text{Re } \hat{\mathbf{x}} \cdot \mathbf{E}$ distributions, $y - z$ plane, $x = 0$.

```

0.01d0
fixed_or_random_orientation
0
calculate_scattering_coefficients
0
scattering_coefficient_file
amn_cylslab.dat
scattering_plane_angle_deg
0.d0
incident_azimuth_angle_deg
0.d0
incident_polar_angle_deg
0.d0
gaussian_beam_constant
0.05d0
gaussian_beam_focal_point
0.d0,0.d0,0.d0
calculate_near_field
1
near_field_plane_coord
1

```

```

near_field_plane_position
0.d0
near_field_plane_vertices
-25.d0,-50.d0,25.d0,0.d0
spacial_step_size
.1d0
polarization_angle_deg
0.d0
near_field_output_file
nf-temp.dat
near_field_output_data
2
plane_wave_epsilon
0.01d0

```

A 3-D representation of the field – in which the z axis is represented by time – is shown in the animation [cyl-z-movie5.avi](#). The animation shows a color density plot of $|\mathbf{E}|^2$ in the $x-y$ planes, with the animation variable being the z position of the plane. Sphere boundaries are denoted by black lines, and they grow and diminish as the plane sweeps through the spheres. The localization and diffusion of the field can be directly observed. The frames for the animation were created by a simple modification of the main program, and this modification is discussed in the next section.

5 Code elements and customization

The prepackaged main program allows for a wide set of calculation and output options, yet many users may prefer to modify the main program to enable a looped set of calculations (i.e., over a set range of sphere size parameters or incident angles) or to generate output not provided by the standard package (such as scattering matrix values over a hemisphere or near field values on a non-rectangular grid). The purpose of this section is to describe the basic structure, calculation sequence, and subroutine calling conventions, with the intention of giving the user sufficient information to modify the code to their own purposes.

5.1 Standard main program structure

The listing for the standard main program is sparsely commented. Given here is a more detailed description of the code. The preamble (declarations) is skipped, yet note that the `implicit none` convention is used throughout and all variables must be explicitly declared.

1. Read input file parameters and options (using either the default input file or that specified from the command line):

```

printinputdata=1
numargs=mstm_nargs()
if(numargs.eq.0) then
  inputfile='mstm.inp'
else
  call mstm_getarg(inputfile)
endif
call inputdata(inputfile,printinputdata)

```

`mstm_nargs` and `mstm_getarg` is aliases for command line number of arguments and retrieval. These functions/subroutines need to be defined appropriate to the user's compiler in `mstm-intrinsics.f90`.

2. Retrieve run and sphere variables, and allocate arrays:

```
call getspheredata(number_spheres=nsphere)
allocate(xsp(nsphere),rpos(3,nsphere),nodr(nsphere),ntran(nsphere), &
         ri(2,nsphere),sphereblk(nsphere),sphereoff(nsphere+1))
call getspheredata(sphere_size_parameters=xsp,sphere_positions=rpos, &
                 sphere_refractive_indices=ri,volume_size_parameter=xv)
call getrunparameters(mie_epsilon=epsmie,translation_epsilon=epstran, &
                    solution_epsilon=epssoln,max_number_iterations=niter, &
                    fixed_or_random_orientation=fixedorrandom,output_file=outfile, &
                    min_scattering_angle_deg=theta1d,max_scattering_angle_deg=theta2d, &
                    number_scattering_angles=numtheta,gaussian_beam_constant=cbeam, &
                    gaussian_beam_focal_point=gbfocus,run_print_unit=runprintunit, &
                    max_memory_per_processor=maxmbperproc, &
                    normalize_scattering_matrix=normalizesm, &
                    store_translation_matrix=storetranmat, &
                    near_field_distance=nfdistance, &
                    iterations_per_correction=niterstep)
if(numtheta.gt.0) then
    allocate(smt(4,4,numtheta))
endif
endif
```

The arrays `xsp`, `rpos`, `ri`, `nodr`, `ntran` are the size parameters, x, y, z positions, L and R refractive indices (`ri(1,i)` and `ri(2,i)`), and sphere-based and target-based order limits for the spheres. The subroutines `getspheredata` and `getrunparameters` retrieve the sphere and run information as read from the input and sphere position files. The following loop determines if optical activity is present – which will be the case when any of the spheres has different L and R refractive indices.

```
nonactive=1
do i=1,nsphere
    if(cdabs(ri(1,i)-ri(2,i)).gt.1.d-10) then
        nonactive=0
        exit
    endif
enddo
```

3. Calculate single-sphere properties (Mie coefficients, expansion order limits, etc.):

```
call miecoefcalc(nsphere,xsp,ri,epsmie)
call getmiedata(sphere_order=nodr,max_order=nodrmax,number_equations=neqns, &
               sphere_block=sphereblk,sphere_block_offset=sphereoff)
```

The arrays `sphereblk`, `sphereoff` are used to define the address of the scattering coefficients for a particular sphere in a single, compacted array containing the scattering coefficients for all the spheres. `neqns` is the number of complex equations in Eq. (5).

4. Initiate the MPI environment and determine the distribution of processors into groups for T matrix calculation:

```

call ms_mpi(mpi_command='init')
call ms_mpi(mpi_command='size',mpi_size=numprocs)
call ms_mpi(mpi_command='rank',mpi_rank=rank)
call mpisetup(nsphere,nodr,rpos,fixedorrandom,maxbperproc,storetranmat, &
             nfdistance,fftranpresent,runprintunit)
call ms_mpi(mpi_command='barrier')
call mpirottranmtrxsetup(nsphere,nodr,rpos,(1.d0,0.d0),storetranmat,&
                        nfdistance,runprintunit)
call ms_mpi(mpi_command='barrier')

```

Precisely what happens in this step depends on whether the code is run on a serial (compiled with `mpidefs-serial.f90`) or a parallel (`mpdefs-parallel.f90`) machine. Note that only aliases of the MPI instructions appear in the main program and the subroutines in `mstm-modules.f90`; the connections between the aliases and the actual MPI instructions are contained in either `mstm-parallel` or `mstm-serial`. When a serial job is run, the MPI instructions basically do nothing except return. The structure of the MPI alias subroutine calls will be discussed in a subsequent section. `numprocs` and `rank` are the number of processors and rank (processor ID) for the run. Subroutine `mpisetup` partitions the processors into groups for solution of Eq. (34). It also determines the near field groups for the hybrid CBGM; `fftranpresent` is an integer (0/1) returned by the subroutine which indicates whether far field translations are present. Subroutine `mpirotranmtrxsetup` calculates and stores the translations matrices for the interaction equations.

5. Determine the orders required to expand the sphere-based expansions about the target origin:

```
call tranorders(nsphere,nodr,rpos,epstran,ntran,nodrt)
```

6. The code now reaches a branch based on the fixed or random orientation option.

If `fixed_or_random_orientation = 1` (random orientation option), then

- 6.a. Calculate the T matrix per the sequential solutions to Eq. (34) and write/append to file, or read the T matrix from the specified file:

```

call getrunparameters(calculate_t_matrix=calctmatrix,t_matrix_file=tmatrixfile, &
                     t_matrix_convergence_epsilon=epstcon)
allocate(qext(nsphere,1), qabs(nsphere,1), qsca(nsphere,1))
if(calctmatrix.ge.1) then
  if(rank.eq.0) time1=mytime()
  call tmatrixsoln(neqns,nsphere,nodr,nodrt,xsp,rpos,epssoln,epstcon,niter,&
                 calctmatrix,tmatrixfile,fftranpresent,niterstep,qext,qabs,qsca,istat)
  if(rank.eq.0) then
    time2=mytime()-time1
    call timewrite(runprintunit,' execution time:',time2)
  endif
  call rottranmtrxclear()
else
  if(rank.eq.0) then
    open(3,file=tmatrixfile)
    read(3,*) nodrt
    close(3)
    write(runprintunit,('( t matrix order:',i5)') nodrt

```

```

        call flush(runprintunit)
    endif
    nodrta(1)=nodrt
    call ms_mpi(mpi_command='bcast',mpi_send_buf_i=nodrta,mpi_number=1,mpi_rank=0)
    nodrt=nodrta(1)
    call ms_mpi(mpi_command='barrier')
endif

```

The subroutine `tmatrixsoln` is called only to calculate elements of the T matrix, either from the beginning or from the next order of a partially-calculated matrix. If `calctmatrix = 0`, indicating the T matrix already exists in the file, the file is opened and the order `nodrt` of the T matrix is read and broadcast to the other processors.

- 6.b. Calculate the random orientation efficiency factors, the scattering matrix expansion coefficients, and the scattering matrix values for the specified range of scattering angles:

```

nblkt=nodrt*(nodrt+2)
nodrg=nodrt*2
allocate(smc(4,4,0:nodrg))
call ranorientscatmatrix(xv,nsphere,nodrt,nodrg,cbeam, &
    tmatrixfile,smc,qext,qabs)
if(rank.eq.0) then
    qexttot=sum(qext(:,1)*xsp*xsp)/xv/xv
    qabstot=sum(qabs(:,1)*xsp*xsp)/xv/xv
    qscatot=qexttot-qabstot
    asymparm=dbl(smc(1,1,1)/smc(1,1,0))/3.d0
    call ranorientscatmatrixcalc(numtheta,theta1d,theta2d,&
        1,smc,nodrg,smt)
endif
endif

```

`smc` is the array of scattering matrix expansion coefficients and `smt` is the $4 \times 4 \times \text{numtheta}$ array of scattering matrix values. Only the upper triangular elements are computed.

7. If `fixed_or_random_orientation = 0` (fixed orientation option), then

- 7.a. Calculate the sphere scattering coefficients per solution to Eq. (5) for the set α , β and write to file, or read sphere scattering coefficients from file:

```

call getrunparameters(calculate_scattering_coefficients=calcamn, &
    scattering_coefficient_file=amnfile, &
    scattering_plane_angle_deg=phideg, &
    incident_azimuth_angle_deg=alphadeg, &
    incident_polar_angle_deg=betadeg, &
    track_iterations=trackiterations)
alpha=alphadeg*pi/180.d0
beta=betadeg*pi/180.d0
phi=phideg*pi/180.d0
allocate(amnp(neqns,2))
allocate(qext(nsphere,3), qabs(nsphere,3), qsca(nsphere,3))
if(calcamn.eq.1) then
    if(rank.eq.0) time1=mytime()
    call fixedorsoln(neqns,nsphere,nodr,alpha,beta,cbeam,xsp,rpos,epssoln,&

```

```

        epstran,niter,amnp,qext,qabs,qsca,maxerr,maxiter,trackiterations, &
        fftranpresent,niterstep,istat)
if(rank.eq.0) then
    time2=mytime()-time1
    write(runprintunit,'('' max iterations, soln error:'' ,i6,e13.5)') &
        maxiter,maxerr
    call timewrite(runprintunit,' execution time:',time2)
    open(3,file=amnfile)
    do i=1,nsphere
        write(3,'(6e13.5)') qext(i,:),qabs(i,:),qsca(i,:)
        allocate(amnp1(0:nodr(i)+1,nodr(i),2),amnp2(0:nodr(i)+1,nodr(i),2))
        ip1=sphereoff(i)+1
        ip2=sphereoff(i)+sphereblk(i)
        amnp1=reshape(amnp(ip1:ip2,1),(/nodr(i)+2,nodr(i),2/))
        amnp2=reshape(amnp(ip1:ip2,2),(/nodr(i)+2,nodr(i),2/))
        do n=1,nodr(i)
            do m=-n,n
                if(m.le.-1) then
                    ma=n+1
                    na=-m
                else
                    ma=m
                    na=n
                endif
                write(3,'(4e17.9)') amnp1(ma,na,1),amnp2(ma,na,1)
                write(3,'(4e17.9)') amnp1(ma,na,2),amnp2(ma,na,2)
            enddo
        enddo
        deallocate(amnp1,amnp2)
    enddo
    close(3)
endif
else
if(rank.eq.0) then
    open(3,file=amnfile)
    do i=1,nsphere
        read(3,'(6e13.5)') qext(i,:),qabs(i,:),qsca(i,:)
        allocate(amnp1(0:nodr(i)+1,nodr(i),2),amnp2(0:nodr(i)+1,nodr(i),2))
        do n=1,nodr(i)
            do m=-n,n
                if(m.le.-1) then
                    ma=n+1
                    na=-m
                else
                    ma=m
                    na=n
                endif
                read(3,'(4e17.9)') amnp1(ma,na,1),amnp2(ma,na,1)
                read(3,'(4e17.9)') amnp1(ma,na,2),amnp2(ma,na,2)
            enddo
        enddo
    enddo
endif
endif

```

```

        enddo
        ip1=sphereoff(i)+1
        ip2=sphereoff(i)+sphereblk(i)
        amnp(ip1:ip2,1)=reshape(amnp1(0:nodr(i)+1,1:nodr(i),1:2),(/sphereblk(i)/))
        amnp(ip1:ip2,2)=reshape(amnp2(0:nodr(i)+1,1:nodr(i),1:2),(/sphereblk(i)/))
        deallocate(amnp1,amnp2)
    enddo
    close(3)
endif
nsend=neqns*2
call ms_mpi(mpi_command='bcast',mpi_send_buf_dc=amnp,mpi_number=nsend,mpi_rank=0)
endif

```

For most applications, the VSWH expansion functions such as a_{mnp} are stored in a rank-3 matrix, with elements given by

$$a_{mnp} = \begin{cases} a(n+1, -m, p), & -n \leq m \leq -1 \\ a(m, n, p), & 0 \leq m \leq n \end{cases} \quad (46)$$

The dimensions of the array a are $a(0:L+1, L, 2)$, where L is the maximum order. The subroutine `fixedorsoln` returns a rank-2 array `amnp(neqns, 2)`, in which the second dimension denotes the incident polarization state (β or α) and the first dimension contains the corresponding scattering coefficients for the spheres, in fortran-convention compacted order.

- 7.b. Calculate the sphere efficiency factors (for parallel and perpendicular polarization to the scattering plane), the common-origin, rotated scattered field expansion, and the scattering matrix values for the set range of scattering angles:

```

cphi=cos(phi)
sphi=sin(phi)
qexttotpar=sum((qext(:,1)*cphi*cphi+2.d0*qext(:,3)*cphi*sphi+qext(:,2)*sphi*sphi) &
    *xsp*xsp)/xv/xv
qexttotper=sum((qext(:,1)*sphi*sphi-2.d0*qext(:,3)*cphi*sphi+qext(:,2)*cphi*cphi) &
    *xsp*xsp)/xv/xv
qabstotpar=sum((qabs(:,1)*cphi*cphi+2.d0*qabs(:,3)*cphi*sphi+qabs(:,2)*sphi*sphi) &
    *xsp*xsp)/xv/xv
qabstotper=sum((qabs(:,1)*sphi*sphi-2.d0*qabs(:,3)*cphi*sphi+qabs(:,2)*cphi*cphi) &
    *xsp*xsp)/xv/xv
qscatotpar=qexttotpar-qabstotpar
qscatotper=qexttotper-qabstotper
qexttot=(qexttotpar+qexttotper)*.5d0
qabstot=(qabstotpar+qabstotper)*.5d0
qscatot=(qscatotpar+qscatotper)*.5d0
qext(:,1)=(qext(:,1)+qext(:,2))* .5d0
qabs(:,1)=(qabs(:,1)+qabs(:,2))* .5d0
qsca(:,1)=(qsca(:,1)+qsca(:,2))* .5d0
call rottranmtxclear()
allocate(amnp0(0:nodrt+1,nodrt,2,2),pmnp0(0:nodrt+1,nodrt,2,2))
do k=1,2
    call amncommonorigin(neqns,nsphere,nodr,ntran,nodrt,rpos, &
        amnp(1:neqns,k),amnp0(0:,: ,1: ,k))

```

```

        call rotvec(alpha,beta,0.d0,nodrt,nodrt,amp0(0:,1:,1:,k),1)
    enddo
    allocate(gmn(0:2))
    call s11expansion(amp0,nodrt,0,1,gmn)
    asymparm=dbl(gmn(1)/gmn(0))/3.d0
    do i=1,numtheta
        thetad=theta1d+(theta2d-theta1d)*(i-1)/max(1.d0,dbl(numtheta-1))
        costheta=cos(thetad*pi/180.d0)
        call scatteringmatrix(amp0,nodrt,xv,costheta,phi,sa,smt(:, :, i))
    enddo
    deallocate(amp0,pmp0,gmn)

```

8. Write calculation results to output file. These steps are straightforward and need no explanation aside from the code list.
9. If `fixed_or_random_orientation = 0` and `calculate_near_field = 1`, calculate the near field values on the specified array and write to the near field output file.

```

if(fixedorrandom.eq.0) call getrunparameters(calculate_near_field=calcnf)
if(fixedorrandom.eq.0.and.calcnf.eq.1) then
    call getrunparameters(near_field_plane_coord=nfplane, &
        near_field_plane_position=nfplanepos,near_field_plane_vertices=nfplanevert, &
        spacial_step_size=deltax,polarization_angle_deg=gammadeg, &
        near_field_output_file=nfoutfile,near_field_output_data=nfoutdata, &
        plane_wave_epsilon=epsnw)
    nfoutunit=2
    if(rank.eq.0) then
        open(nfoutunit,file=nfoutfile)
    endif
    gamma=gammadeg*pi/180.d0
    call nearfieldgridcalc(neqns,nsphere,nodr,alpha,beta,cbeam,xsp,rpos,ri,amp, &
        nfplane,nfplanepos,nfplanevert,deltax,gamma,nfoutunit,epsnw, &
        nfoutdata,runprintunit)
    if(rank.eq.0) then
        close(nfoutunit)
    endif
endif
endif

```

5.2 MPI alias instructions

To facilitate the compilation and execution of the code on both serial and parallel machines, the main program and the subroutine modules employ aliases to the standard MPI instruction set. The association of the aliases with the MPI commands is performed solely within `mpidefs-serial.f90` or `mpidefs-parallel.f90`. The general format of the call to the alias subroutine is as follows

```

call ms_mpi(mpi_command=command,mpi_recv_buf_type=recvbuf, &
    mpi_send_buf_type=sendbuf,mpi_number=number, &
    mpi_comm=comm,mpi_group=group,mpi_rank=rank, &
    mpi_size=size,mpi_new_comm=newcom, &
    mpi_new_group=newgroup,mpi_new_group_list=newgrouplist, &
    mpi_operation=operation)

```

All of the arguments are optional with the exception of `mpi_command`, and usage of the subroutine involves the keyword format as shown above. The arguments are as follows:

`mpi_command`: Character string denoting MPI command. Options are 'init', 'finalize', 'size', 'rank', 'group', 'incl', 'create', 'barrier', 'bcast', 'send', 'recv', 'reduce', and 'allreduce', and the commands correspond directly to the associated MPI standard instructions.

`mpi_send_buf_type`: Send buffer for MPI operations. Options for *type* are `i` (integer), `r` (single precision real), `c` (single precision complex), `dp` (double precision), and `dc` (double precision complex). If the command is either 'reduce' or 'allreduce', and `mpi_send_buf_type` is not present in the argument list, the subroutine defaults to `mpi_send_buf_type = mpi_in_place`. The variable associated with the buffer must be an array of at least rank 1; if you need to send a single scalar element, you need to equate it to a scratch array of rank 1 and length 1. For example,

```
integer :: nodrt
call ms_mpi(mpi_command='bcast',mpi_send_buf_i = nodrt, ...
```

will likely produce a compilation error. The way to do it is

```
integer :: nodrt, nodrt_temp(1)
nodrt_temp(1)=nodrt
call ms_mpi(mpi_command='bcast',mpi_send_buf_i = nodrt_temp, ...
nodrt=nodrt_temp(1)
```

`mpi_recv_buf_type`: Receive buffer for MPI operations. Options for *type* are the same as above.

`mpi_number`: Number of buffer elements sent/received. Integer.

`mpi_comm`: MPI communicator, integer. Default (if not present) is `mpi_comm_world`.

`mpi_group`: MPI group, integer.

`mpi_rank`: MPI rank, integer.

`mpi_size`: MPI size (number of processors in group and/or communicator), integer.

`mpi_new_comm`: MPI communicator handle for new group creation, integer. Associated with 'create' command.

`mpi_new_group`: MPI group ID for new group creation, integer. Associated with 'incl' command

`mpi_new_group_list`: Listing of processor ranks for new group , integer rank 1 array. Associated with 'incl' command.

`mpi_operation`: Operation command for 'reduce' and 'allreduce' commands. Options are `ms_mpi_max`, `ms_mpi_min`, and `ms_mpi_sum`. These three variables are integers and are defined globally once the 'init' command has been called, so the sum operation would be invoked by `mpi_operation = ms_mpi_sum`, as written.

Observe that there is no `mpi_type` argument to the `ms_mpi` subroutine. The buffer type is now implicit in the type of the send and/or receive buffer, i.e., `i`, `r`, etc.

5.3 Code modification and customizing

The user with average fortran knowledge should be able to modify the main program to suit their specialized computational needs. As an aid to such projects, a modified main program `mstm-generic-main.f90` has been provided in the distribution, and is intended to serve as a template for user customization. The generic main eliminates the command line retrieval and input file operations, and all calculation variables and parameters are hardwired (i.e., explicitly defined) in the code. The code employs a simple `do - enddo` loop to perform repeated calculations over a set parameter range. The basic form of this main program was used to generate the animation frames in the examples shown in `mstm_gallery.pdf`.

6 Revisions

1.2 (21 February 2011)

- The normalization of the phase function S_{11} was changed to

$$\frac{1}{2} \int_0^\pi S_{11}(\theta) \sin \theta d\theta = 1$$

- Fixed orientation results now output the efficiency factors for parallel and perpendicular polarization.
- A bug was fixed in the calculation of the fixed orientation scattering matrix. Additional bug fixes for calling of `flush` subroutine and opening/closing the input file.
- An appendix was added to the manual, for defining the definitions of the various mathematical quantities. This appendix will expand with time.

2.1 (30 March 2011)

- The formulation was extended to include optically active spheres. This was more of a major than a minor modification; roughly 1/3 of the subroutines needed alteration to account for the presence of left and right refractive index values. Optical activity results in a very minor slowdown in performance, when compared against calculations done using a non-active code; this is due to the added complexity in iterating Eq. (5) as a result of the TE and TM mode coupling for optically active spheres.
- Miscellaneous input and output options were added, in response to user suggestions.

2.2 (11 November 2011)

- The hybrid CBGM was installed to limit memory use without sacrificing execution time; this modification was in response to suggestions receive from a number of users. In addition, a new recurrence formula was added to calculate the translation matrix elements for axial translation; this is described in the mathematical section.

Appendix: Mathematical relations and definitions

The purpose of this section is to identify the various mathematical quantities appearing in the formulation, and to link the quantities with their coding. Two resources will be used for coding, being 1) the MSTM Fortran-90 subroutines and functions, and 2) the *Mathematica* symbolic mathematics package. More often than not, the definitions of functions presented here will not entirely line up with "traditional" or "established" definitions. In particular, the normalizations used for harmonic functions can be different than reported elsewhere.

Optically active formulation

The defining characteristic of OA media is that only circularly polarized, homogenous plane waves can propagate in such media without undergoing a change in polarization state. Following the formulation of Bohren and Huffman [8], the electric and magnetic fields propagating within OA media can be constructed from a linear transformation of left and right expansion functions[8], denoted as Q_L and Q_R , i.e,

$$\begin{pmatrix} \mathbf{E} \\ \mathbf{H} \end{pmatrix} = \mathbf{A} \begin{pmatrix} Q_L \\ Q_R \end{pmatrix} \quad (47)$$

in which the matrix \mathbf{A} is

$$\mathbf{A} = \begin{pmatrix} 1 & -i/m \\ -i m & 1 \end{pmatrix} \quad (48)$$

The \mathbf{Q} expansion functions satisfy

$$\nabla \times \mathbf{Q}_L = k_L \mathbf{Q}_L \quad (49)$$

$$\nabla \times \mathbf{Q}_R = -k_R \mathbf{Q}_R \quad (50)$$

in which $k_L = 2\pi m_L/\lambda$ and $k_R = 2\pi m_R/\lambda$ are the left and right wavenumbers for the medium, with m_L and m_R denoting the relative refractive indices for left and right circularly polarized light. The left and right refractive indices are related to the average (or bulk) refractive index m by

$$m = \frac{2}{1/m_L + 1/m_R} \quad (51)$$

In addition, the left and right refractive indices can be specified by the average m and a complex chiral factor χ by

$$m_L = \frac{m}{1 - \chi m}, \quad m_R = \frac{m}{1 + \chi m} \quad (52)$$

The \mathbf{Q} expansion functions can be represented in the interior of an OA sphere by expansions of the regular VSWF $\mathbf{N}_{mnp}^{(1)}(\mathbf{kr})$, of order n , degree m , and mode $p = 1$ (TM) or 2 (TE). The VSWFs satisfy the vector Helmholtz equation

$$\nabla^2 \mathbf{N}_{mnp}^{(1)}(\mathbf{kr}) + k^2 \mathbf{N}_{mnp}^{(1)}(\mathbf{kr}) = 0 \quad (53)$$

as well as

$$\nabla \times \mathbf{N}_{mn1}^{(1)}(\mathbf{kr}) = k \mathbf{N}_{mn2}^{(1)}(\mathbf{kr}), \quad \nabla \times \mathbf{N}_{mn2}^{(1)}(\mathbf{kr}) = k \mathbf{N}_{mn1}^{(1)}(\mathbf{kr}) \quad (54)$$

With this set of basis functions, formulas for the \mathbf{Q} expansion functions will appear as

$$\mathbf{Q}_L(\mathbf{r}) = \sum_{n=1} \sum_{m=-n}^n c_{mnL} \left(\mathbf{N}_{mn1}^{(1)}(\mathbf{k}_L \mathbf{r}) + \mathbf{N}_{mn2}^{(1)}(\mathbf{k}_L \mathbf{r}) \right) \quad (55)$$

$$\mathbf{Q}_R(\mathbf{r}) = \sum_{n=1} \sum_{m=-n}^n c_{mnR} \left(\mathbf{N}_{mn1}^{(1)}(\mathbf{k}_R \mathbf{r}) - \mathbf{N}_{mn2}^{(1)}(\mathbf{k}_R \mathbf{r}) \right) \quad (56)$$

in which c_{mnL} and c_{mnR} denote the sought expansion coefficients for the internal field.

Exterior to the sphere, the incident and scattered fields will be represented by the usual expansions of regular and outgoing VSWF, i.e.,

$$\mathbf{E}_{inc}(\mathbf{r}) = \sum_{n=1} \sum_{m=-n}^n \sum_{p=1}^2 f_{mnp} \mathbf{N}_{mnp}^{(1)}(\mathbf{k} \mathbf{r}) \quad (57)$$

$$\mathbf{E}_{sca}(\mathbf{r}) = \sum_{n=1} \sum_{m=-n}^n \sum_{p=1}^2 a_{mnp} \mathbf{N}_{mnp}^{(3)}(\mathbf{k} \mathbf{r}) \quad (58)$$

in which the expansion coefficients f for the incident field will depend on the direction and polarization state of the field. Evaluation of the continuity conditions at the sphere surface will yield a linear relation between the scattering and incident field coefficients, of the form

$$a_{mnp} = \sum_{q=1}^2 \bar{a}_{n;pq} f_{mnq} \quad (59)$$

in which the coefficients \bar{a} will be a function of the sphere size parameter ka and the left and right refractive indices. These are analogous to the Lorenz–Mie coefficients for non active spheres. Explicit formulas are

$$\bar{a}_{n;11} = - \frac{V_n(L)A_n(R) + V_n(R)A_n(L)}{D_n} \quad (60)$$

$$\bar{a}_{n;22} = - \frac{W_n(L)B_n(R) + W_n(R)B_n(L)}{D_n} \quad (61)$$

$$\bar{a}_{n;12} = \bar{a}_{n;21} = \frac{W_n(L)A_n(R) + W_n(R)A_n(L)}{D_n} \quad (62)$$

$$D_n = W_n(L)V_n(R) + W_n(R)V_n(L) \quad (63)$$

$$W_n(J) = - (\xi_n(x) \psi'_n(x \mathbf{m}_J)) + \mathbf{m} \psi_n(x \mathbf{m}_J) \xi'_n(x) \quad (64)$$

$$V_n(J) = - (\mathbf{m} \xi_n(x) \psi'_n(x \mathbf{m}_J)) + \psi_n(x \mathbf{m}_J) \xi'_n(x) \quad (65)$$

$$A_n(J) = \mathbf{m} \psi_n(x \mathbf{m}_J) \psi'_n(x) - \psi_n(x) \psi'_n(x \mathbf{m}_J) \quad (66)$$

$$B_n(J) = \psi_n(x \mathbf{m}_J) \psi'_n(x) - \mathbf{m} \psi_n(x) \psi'_n(x \mathbf{m}_J) \quad (67)$$

in which ψ_n and ξ_n denote the Ricatti–Bessel and Ricatti–Hankel functions. The coefficients for the internal fields can be related to the scattering coefficients, again by application of the surface continuity conditions.

The formulas are

$$c_{nmL} = \bar{c}_{n;L1} a_{mn1} + \bar{c}_{n;L2} a_{mn2} \quad (68)$$

$$c_{nmR} = \bar{c}_{n;R1} a_{mn1} + \bar{c}_{n;R2} a_{mn2} \quad (69)$$

$$\bar{c}_{n;L1} = -\frac{i m_L B_n(R)}{F_n} \quad (70)$$

$$\bar{c}_{n;L2} = -\frac{i m_L A_n(R)}{F_n} \quad (71)$$

$$\bar{c}_{n;R1} = \frac{m_R m B_n(L)}{F_n} \quad (72)$$

$$\bar{c}_{n;R2} = -\frac{m_R m A_n(L)}{F_n} \quad (73)$$

$$F_n = A_n(L)B_n(R) + A_n(R)B_n(L) \quad (74)$$

The most salient feature of the single-sphere formulation, with respect to the extension of the formulation to multiple spheres, is that the single-sphere TE and TM modes are coupled for optically active spheres. That is, a TM mode incident field excitation will produce a TE mode scattering component, and visa-versa. This is in contrast to Lorenz-Mie scattering, for which the excitation and scattering modes are uncoupled.

Vector Spherical Wave Functions

The VSWF used in the formulation and the code are defined by

$$\mathbf{N}_{mn2}^{(\nu)}(\mathbf{r}) = \left(\frac{2}{n(n+1)} \right)^{1/2} \nabla \times \left(\mathbf{r} \psi_{mn}^{(\nu)}(\mathbf{r}) \right) \quad (75)$$

$$\mathbf{N}_{mn1}^{(\nu)}(\mathbf{r}) = \frac{1}{k} \nabla \times \mathbf{N}_{mn2}^{(\nu)}(\mathbf{r}) \quad (76)$$

where ψ denotes the scalar wave function;

$$\psi_{mn}^{(\nu)}(\mathbf{r}) = \begin{cases} j_n(kr) Y_{mn}(\cos \theta, \phi) & \nu = 1 \\ h_n(kr) Y_{mn}(\cos \theta, \phi) & \nu = 3 \end{cases} \quad (77)$$

with j_n and $h_n = j_n + i y_n$ representing the spherical Bessel and Hankel functions and Y_{mn} denoting the spherical harmonic,

$$Y_{mn}(\cos \theta, \phi) = \left(\frac{2n+1}{4\pi} \frac{(n-m)!}{(n+m)!} \right)^{1/2} P_n^m(\cos \theta) e^{im\phi} \quad (78)$$

where P_n^m is the Associated Legendre function.

The cartesian components of the VSWF are given as

$$\begin{aligned} (\hat{\mathbf{x}} + i\hat{\mathbf{y}}) \cdot \mathbf{N}_{mn1}^{(\nu)}(\mathbf{r}) = & - \left(\frac{2}{n(n+1)(2n+1)} \right)^{1/2} \left[n \left(\frac{(n+m+1)(n+m+2)}{2n+3} \right)^{1/2} \psi_{m+1n+1}^{(\nu)}(\mathbf{r}) \right. \\ & \left. - (n+1) \left(\frac{(n-m)(n-m-1)}{2n-1} \right)^{1/2} \psi_{m+1n-1}^{(\nu)}(\mathbf{r}) \right] \end{aligned} \quad (79)$$

$$\begin{aligned} (\hat{\mathbf{x}} - i\hat{\mathbf{y}}) \cdot \mathbf{N}_{mn1}^{(\nu)}(\mathbf{r}) = & \left(\frac{2}{n(n+1)(2n+1)} \right)^{1/2} \left[n \left(\frac{(n-m+1)(n-m+2)}{2n+3} \right)^{1/2} \psi_{m-1n+1}^{(\nu)}(\mathbf{r}) \right. \\ & \left. + (n+1) \left(\frac{(n+m)(n+m-1)}{2n-1} \right)^{1/2} \psi_{m-1n-1}^{(\nu)}(\mathbf{r}) \right] \end{aligned} \quad (80)$$

$$\begin{aligned} \hat{\mathbf{z}} \cdot \mathbf{N}_{mn1}^{(\nu)}(\mathbf{r}) = & \left(\frac{2}{n(n+1)(2n+1)} \right)^{1/2} \left[n \left(\frac{(n+m+1)(n-m+1)}{2n+3} \right)^{1/2} \psi_{mn+1}^{(\nu)}(\mathbf{r}) \right. \\ & \left. + (n+1) \left(\frac{(n+m)(n-m)}{2n-1} \right)^{1/2} \psi_{mn-1}^{(\nu)}(\mathbf{r}) \right] \end{aligned} \quad (81)$$

$$(\hat{\mathbf{x}} + i\hat{\mathbf{y}}) \cdot \mathbf{N}_{mn2}^{(\nu)}(\mathbf{r}) = -i \left(\frac{2(n-m)(n+m+1)}{n(n+1)} \right)^{1/2} \psi_{m+1n}^{(\nu)}(\mathbf{r}) \quad (82)$$

$$(\hat{\mathbf{x}} - i\hat{\mathbf{y}}) \cdot \mathbf{N}_{mn2}^{(\nu)}(\mathbf{r}) = -i \left(\frac{2(n+m)(n-m+1)}{n(n+1)} \right)^{1/2} \psi_{m-1n}^{(\nu)}(\mathbf{r}) \quad (83)$$

$$\hat{\mathbf{z}} \cdot \mathbf{N}_{mn2}^{(\nu)}(\mathbf{r}) = -im \left(\frac{2}{n(n+1)} \right)^{1/2} \psi_{mn}^{(\nu)}(\mathbf{r}) \quad (84)$$

These functions are calculated in the subroutine

```
subroutine vwhcalc(rpos,ri,nodr,itYPE,vwh)
```

and inspection of the subroutine will reveal the roles of the arguments.

Wigner D functions

The definition of the $\mathcal{D}_{mk}^{(n)}$ functions used in the code is

$$\mathcal{D}_{mk}^{(n)}(x) = (-1)^{m+k} \left[\frac{(n-k)!(n+k)!}{(n-m)!(n+m)!} \left(\frac{1+x}{2} \right)^{m+k} \left(\frac{1-x}{2} \right)^{k-m} \right]^{1/2} P_{n-k}^{(k-m,k+m)}(x) \quad (85)$$

where $P_{n-k}^{(k-m,k+m)}(x)$ is the Jacobi Polynomial. The *Mathematica* function definition for the $\mathcal{D}_{mk}^{(n)}$ function is

```
drot[m_, n_, k_, x_] := (-1)^(m+k) ((n-k)!(n+k)!/(n-m)!(n+m)!)^(1/2)
((1+x)/2)^(m+k)/2 ((1-x)/2)^(k-m)/2 JacobiP[n-k, k-m, k+m, x]
```

The functions are calculated in MSTM via recurrence relations in the subroutine

subroutine rotcoef(cbe,kmax,nmax,dc)

which returns an array for $|k| \leq \mathbf{kmax}$ and $n = 0, 1, \dots, \mathbf{nmax}$, $|m| \leq n$. The function argument $\mathbf{cbe} = x$. The addressing convention is

$$\mathbf{dc}(\mathbf{k}, \mathbf{n} * (\mathbf{n} + 1) + \mathbf{m}) = \mathcal{D}_{km}^{(n)}$$

Translation matrix elements

The explicit formula for the translation matrix elements is

$$J_{mnpklq}(\mathbf{r}) = -(-1)^m [4\pi(2n+1)(2l+1)]^{1/2} i^{n-l} \sum_{|n-l|+|p-q|, 2}^{n+l-|p-q|} \frac{i^w}{(2w+1)^{1/2}} C_{-mn\ kl}^w C_{-1n\ 1l}^w \psi_{k-m\ w}^{(1)}(\mathbf{r}) \quad (86)$$

The "2" appearing in the summation limit denotes that w runs over every other value, starting with the lower and ending with the upper limit. The formula for H is identical, except that the outgoing wave function is used. The C quantities are shorthand for the Clebsch–Gordan coefficients,

$$C_{mn\ kl}^w = C((n, m), (l, k), (w, m+k)) \quad (87)$$

The *Mathematica* definition of the CG coefficients is simply

`cc[m_, n_, k_, l_, w_] := ClebschGordan[{n, m}, {l, k}, {w, m+k}]`

The MSTM code uses a recursive procedure to calculate the CG coefficients, employing both downward and upwards recursion.

Recurrence formula for axial translation

The regular translation matrix obey the following relations;

$$e^{i\mathbf{k}\cdot\mathbf{r}} \mathbf{\Pi}_{mnp}(\cos\beta, \alpha) = \sum_{l=1}^{\infty} \sum_{k=-l}^l \sum_{q=1}^2 J_{klq\ mnp}(\mathbf{r}) \mathbf{\Pi}_{klq}(\cos\beta, \alpha) \quad (88)$$

$$e^{i\mathbf{k}\cdot\mathbf{r}} \mathbf{\Pi}_{mnp}^*(\cos\beta, \alpha) = \sum_{l=1}^{\infty} \sum_{k=-l}^l \sum_{q=1}^2 J_{mnp\ klq}(\mathbf{r}) \mathbf{\Pi}_{klq}^*(\cos\beta, \alpha) \quad (89)$$

where $\mathbf{\Pi}_{mnp}$ are the vector spherical harmonic functions, and \mathbf{k} is a vector pointed in the β, α direction. For an axial translation $\mathbf{k} \cdot \mathbf{r} = \cos\beta kz$, and azimuth modes become decoupled. Recurrence formulas can be obtained by differentiating the axial translation formulas with respect to kz and applying recurrence formulas for the vector spherical harmonic formulas. This results in the 'diamond'-type recurrence formula;

$$c_{mn} J_{mn+1pmlq} = a_{mn} J_{mn-1pmlq} + i(b_{ml} - b_{mn}) J_{mnpml3-q} + a_{ml} J_{mnpml-1q} - c_{ml} J_{mnpml+1q} \quad (90)$$

with

$$a_{ml} = \left(\frac{(l-1)(l+1)(l-m)(l+m)}{(2l-1)(2l+1)l^2} \right)^{1/2}$$

$$b_{ml} = \frac{m}{l(l+1)}$$

$$c_{ml} = \left(\frac{l(l+2)(l+1-m)(l+1+m)}{(2l+1)(2l+3)(l+1)^2} \right)^{1/2}$$

The formula in Eq. (90) applies equally to the outgoing translation matrix H . Terms in Eq. (90) corresponding to elements with order $<$ degree are implicitly zero. The recurrence uses the explicit formula in Eq. (86) to calculate the boundary elements of $J_{mnp\ m|q}$ and $J_{mnp\ mLq}$ for $n = |m|, |m| + 1, \dots, L$ (excluding $n = 0$, of course), and the recurrence formula is then applied to calculate the remaining elements. This allows the matrix elements to be calculated in order L^3 operations, as opposed to order L^4 required from Eq. (86).

Vector spherical harmonics

The vector spherical harmonic (VSH) functions are defined by

$$\mathbf{\Pi}_{mn1}(\cos\theta, \phi) = (-i)^{n+1} r \left(\frac{1}{n(n+1)} \right)^{1/2} \nabla Y_{mn}(\cos\theta, \phi) \quad (91)$$

$$\mathbf{\Pi}_{mn2}(\cos\theta, \phi) = i \hat{\mathbf{r}} \times \mathbf{\Pi}_{mn1}(\cos\theta, \phi) \quad (92)$$

and are functionally given by

$$\mathbf{\Pi}_{mnp}(\cos\theta, \phi) = (-i)^{n+1} (\tau_{mnp}(\cos\theta) \hat{\mathbf{e}}_\theta + i\tau_{mn3-p}(\cos\theta) \hat{\mathbf{e}}_\phi) e^{im\phi} \quad (93)$$

$$\tau_{mn1}(x) = -(1-x^2)^{1/2} \left(\frac{2n+1}{4\pi n(n+1)} \frac{(n-m)!}{(n+m)!} \right)^{1/2} \frac{dP_n^m(x)}{dx} \quad (94)$$

$$\tau_{mn2}(x) = m (1-x^2)^{-1/2} \left(\frac{2n+1}{4\pi n(n+1)} \frac{(n-m)!}{(n+m)!} \right)^{1/2} P_n^m(x) \quad (95)$$

References

- [1] D. W. Mackowski, Calculation of total cross sections of multiple sphere clusters, *J. Opt. Soc. Amer. A* 11 (1994) 2851–2861. [PDF](#)
- [2] K. A. Fuller, D. W. Mackowski, Electromagnetic scattering by compounded spherical particles, in: M. I. Mishchenko, J. W. Hovenier, L. D. Travis (Eds.), *Light Scattering by Nonspherical Particles: Theory, Measurements, and Applications*, Academic Press, 2000, Ch. 8, p. 226.
- [3] P. Flatau, Fast solvers for one dimensional light scattering in the discrete dipole approximation, *Opt. Express* 12 (2004) 3149–3155.
- [4] K. A. Fuller, Optical resonances and two-sphere systems, *Appl. Opt.* 30 (33) (1991) 4716–4731.
- [5] M. I. Mishchenko, L. D. Travis, A. A. Lacis, *Multiple Scattering of Light by Particles: Radiative Transfer and Coherent Backscattering*, Cambridge University Press, 2006.

- [6] A. Doicu, T. Wriedt, Computation of the beam–shape coefficients in the generalized Lorenz–Mie theory by using the translational addition theorem for spherical vector wave functions, *Appl. Opt.* 13 (1997) 2971–2978.
- [7] A. Doicu, T. Wriedt, Plane wave spectrum of electromagnetic beams, *Opt. Comm.* 136 (1997) 114–124.
- [8] C. F. Bohren, D. R. Huffman, *Absorption and Scattering of Light by Small Particles*, Wiley, 1983.
- [9] D. W. Mackowski, M. I. Mishchenko, Calculation of the T matrix and the scattering matrix for ensembles of spheres, *J. Opt. Soc. Amer. A* 13 (1996) 2266–2278. [PDF](#)
- [10] www.eng.auburn.edu/admin/ens/hpcc/.