ELSEVIER

Journal of Quantitative Spectroscopy & Radiative Transfer 106 (2007) 417–436

Journal of Quantitative Spectroscopy & Radiative Transfer

www.elsevier.com/locate/jqsrt

Comparison between discrete dipole implementations and exact techniques

Antti Penttilä[a,*], Evgenij Zubko[a,b,c], Kari Lumme[a], Karri Muinonen[a], Maxim A. Yurkin[d,e], Bruce Draine[f], Jussi Rahola[g], Alfons G. Hoekstra[d], Yuri Shkuratov[b]

[a]*Observatory, University of Helsinki, P.O. box 14, FI-00014 University of Helsinki, Finland*
[b]*Astronomical Institute of Kharkov National University, 35 Sumskaya Street, Kharkov 61022, Ukraine*
[c]*Institute of Low Temperature Science, Hokkaido University, Kita-ku North 19 West 8, Sapporo 060-0819, Japan*
[d]*Faculty of Science, Section Computational Science of the University of Amsterdam, Kruislaan 403, 1098 SJ, Amsterdam, The Netherlands*
[e]*Institute of Chemical Kinetics and Combustion, Siberian Branch of the Russian Academy of Sciences, Institutskaya Str. 3, 630090 Novosibirsk, Russia*
[f]*Department of Astrophysical Sciences, Princeton University, 108 Peyton Hall, Princeton, NJ 08544-1001, USA*
[g]*Simulintu Oy, Espoo, Finland*

Abstract

The use of the discrete dipole approximation (DDA) method in wave optical scattering simulations is growing quite fast. This is due to the fact that the current computing resources allow to apply DDA to sufficiently large scattering systems. The advantage of DDA is that it is applicable to arbitrary particle shape and configuration of particles. There are several computer implementations of the DDA method, and in this article we will compare four of such implementations in terms of their accuracy, speed and usability. The accuracy is studied by comparing the DDA results against results from either Mie, T-matrix or cluster T-Matrix codes with suitable geometries. It is found that the relative accuracy for intensity is between 2% and 6% for ice and silicate type refractive indices and the absolute accuracy for linear polarization ratio is roughly from 1% to 3%.

Keywords: Scattering; Discrete-dipole approximation

1. Introduction

The importance of light scattering methods in studying the structure of various remotely sensed objects, e.g. in astronomy and in some technological applications has greatly increased in the last years. One obvious reason for this is undoubtedly the enormous and steady increase in the computer power and speed. The information content from the light scattering studies is very large because at its best all the 16 Mueller matrix

*Corresponding author. Fax: +358 9 19122952.
E-mail address: Antti.I.Penttila@helsinki.fi (A. Penttilä).

0022-4073/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jqsrt.2007.01.026

elements as a function of the scattering angle can be derived and related to some unknown physical parameters.

Before light scattering results for any object can be computed the relevant morphology for it must either be known or modeled. Most of the geometries can be divided into two basic classes: solid and particulate. For the solids it is fairly straightforward to model the outer boundary while in the case of particulate material much more free parameters must be considered. First, the sizes and shapes of the constituents must be decided. Second, the packing density (1-porosity) is often a crucial parameter and must either be known or varied which complicates the computations further. For any of existing light scattering codes we must know (or model) the refractive index and in most cases both the real and imaginary parts. The scattering material can also be anisotropic which requires that two or three different indices should be obtained.

Most of the existing light scattering programs can roughly be divided into two categories: those which use the ray optics (RO) or geometric optics (GO) approaches and those which use the exact wave optics (WO) approach. Following Mishchenko et al. [1], we can say that the RO and GO use only the ladder diagrams e.g. the radiative transfer theory (RTT) [2–4], while e.g. the coherent backscattering (CB) uses also the cyclical diagrams.

In the case of the problems which deal with particulate media one of the crucial parameters is the packing density (pD). The classical assumption is that the light scattering by these can be explained by the much simpler RO/GO models if the pD is less that about 5–10%. With higher values of pD the exact WO approach should be used. Unfortunately, up to quite recently the existing light scattering routines and the computers have not been able to do that.

Among the WO methods the discrete dipole approximation (DDA), also known as the coupled dipole approximation, has several special advantages over all the other existing approaches. These are that they can be applied to quite arbitrary shaped geometries which can be inhomogeneous and anisotropic. As already explicitly in its name the DDA has a minor drawback in being an approximation although if infinite CPU time would be possible the results should become exact. The other drawback comes from the fact that if orientation averages are needed then the computationally demanding linear equations must be solved repeatedly. Recent developments in DDA computations include, for instance, the optimization of the DDA computations for scatterers that have identical shapes but differing sizes or refractive indices [5].

Because of the increasing popularity in the DDA codes it would be valuable to quantitatively compare several aspects of these both in relation to each other (speed, accuracy, etc.) and in respect to the absolute accuracy of those. For this we naturally need codes which can handle some geometries in a numerically exact manner. This limits the available geometries to only a few. A fairly obvious choice is to use the T-matrix [6] and the cluster T-matrix [7] codes. We were able to collect a total of four different DDA versions to do this comparison.

This article has been arranged such that in Section 2 we briefly describe the theory of the DDA approach, in Section 3 the four DDA codes are presented, and in Section 4 the details of the comparison and its results are discussed. In Section 5 we draw some conclusions about the performance and properties of different DDA codes.

2. Theory of DDA

Assume that the scattering particles are isotropic and homogeneous, i.e., that their optical properties are fully described by the complex refractive index m. In DDA, the particle is divided into a discrete set of dipoles (or computational cells) on a cubic lattice with dimensionless interdipole distances kd (k is the wave number in free space). The electric field values $\mathbf{E}_j$ ($j=1,2,3,\ldots,N$) are to be derived from a group of linear equations (see e.g. Lumme and Rahola [8]):

$$\mathbf{E}_j = \gamma \mathbf{E}_{0,j} + \beta \sum_{l=1, l\neq j}^{N} \mathbf{T}_{jl} \cdot \mathbf{E}_l, \quad j = 1,2,3,\ldots,N, \tag{1}$$

where $\mathbf{E}_{0,j}$ is the incident field value at dipole j, γ and β are complex coefficients (see below), and $\mathbf{T}_{jl}$ is the transformation dyadic ($\mathbf{1}$ being the unit dyadic)

$$\mathbf{T}_{jl} = u_{jl}\mathbf{1} + v_{jl}\mathbf{e}_{jl}\mathbf{e}_{jl}, \tag{2}$$

where

$$u_{jl} = \frac{\exp(i\rho_{jl})}{\rho_{jl}^3}(\rho_{jl}^2 + i\rho_{jl} - 1),$$
$$v_{jl} = \frac{\exp(i\rho_{jl})}{\rho_{jl}^3}(-\rho_{jl}^2 - i3\rho_{jl} + 3),$$
$$\rho_{jl} = k|\mathbf{r}_j - \mathbf{r}_l|,$$
$$\mathbf{e}_{jl} = \frac{\mathbf{r}_j - \mathbf{r}_l}{|\mathbf{r}_j - \mathbf{r}_l|}, \tag{3}$$

$\mathbf{r}_j$ describing the location of the jth dipole. The incident field is taken to be a plane wave with wave vector $\mathbf{k}$,

$$\mathbf{E}_{0,j} = \mathbf{E}_0 \exp(i\mathbf{k} \cdot \mathbf{r}_j). \tag{4}$$

Following Draine and Goodman [9], γ and β depend on the refractive index and computational cell according to the following equations, where E_0 is assumed to be real-valued (the incident field being linearly polarized in Eq. (4)):

$$\gamma = 1,$$
$$\beta = k^3 \frac{\alpha_{\mathrm{nr}}}{1 - \frac{2}{3}ik^3\alpha_{\mathrm{nr}}},$$
$$\alpha_{\mathrm{nr}} = \frac{\alpha_0}{1 + \alpha_0/d^3(b_1 + m^2 b_2 + m^2 b_3 S)k^2 d^2},$$
$$\alpha_0 = \frac{3d^3}{4\pi}\frac{m^2 - 1}{m^2 + 2},$$
$$b_1 = -1.8915316, \quad b_2 = 0.1648469, \quad b_3 = -1.7700004,$$
$$S = \frac{1}{k^2 E_0^2}(k_x^2 E_{0,x}^2 + k_y^2 E_{0,y}^2 + k_z^2 E_{0,z}^2). \tag{5}$$

A related approach to the DDA is the use of the volume integral equation method (VIEM) (see e.g. Lakhtakia and Mulholland [10] or Rahola [11]). The integral equation for the electric field inside a particle $\mathbf{E}(\mathbf{r})$ is given by:

$$\mathbf{E}(\mathbf{r}) = \mathbf{E}^{\mathrm{inc}}(\mathbf{r}) + k^3 \int_V (m(\mathbf{r}')^2 - 1)\mathbf{G}(\mathbf{r}, \mathbf{r}') \cdot \mathbf{E}(\mathbf{r}')\, \mathrm{d}^3\mathbf{r}', \tag{6}$$

where $\mathbf{E}^{\mathrm{inc}}$ is the incident field, $\mathbf{G}$ is the dyadic Green's function

$$\mathbf{G}(\mathbf{r}, \mathbf{r}') = \left(\mathbf{1} + \frac{\nabla\nabla}{k^2}\right) g(|\mathbf{r} - \mathbf{r}'|) \tag{7}$$

and

$$g(r) = \frac{\mathrm{e}^{ikr}}{4\pi kr}. \tag{8}$$

When piecewise constant basis functions are used in VIEM, the system of linear equations is of the same mathematical form as in DDA, but the coefficients γ and β obtain the following values:

$$\gamma = \frac{1}{1 - (m^2 - 1)(k^3 M - \frac{1}{3})},$$
$$\beta = k^3 d^3 \gamma (m^2 - 1),$$
$$k^3 M = \frac{2}{3}\left[\left(1 - i\sqrt[3]{\frac{3}{4\pi}}kd\right)\exp\left(i\sqrt[3]{\frac{3}{4\pi}}kd\right) - 1\right]. \tag{9}$$

The solution of the group of linear equations allows the computation of scattering matrices as well as scattering, absorption and extinction cross sections pertaining to a given problem (see e.g. Zubko et al. [12] or Lumme and Rahola [13]). The volume integral equation approach allows the use of higher order basis functions which give more accurate solutions for the same computational cells than the DDA approach or the VIEM with piecewise constant basis functions.

3. DDA codes

Several researchers or groups have implemented a DDA code for their own use. Some codes are publicly available, either freely or commercially. In this article we compare four DDA implementations: the commercially available SIRRI, the publicly available DDSCAT and ADDA, and ZDD which is currently available for its developers. The codes are presented in more detail in the following subsections.

3.1. SIRRI

The SIRRI code is based on the VIEM approach and has been developed by Simulintu Oy in Finland. The code has been written in Fortran 90. The geometry of the scatterer is read from a file and all the parameters of the computation are described in a parameter file which has flexible keyword syntax. The code uses dynamic memory allocation so that no recompilations are needed for different problem sizes. Linear equations for isotropic particles are complex symmetric and can be solved with the complex symmetric version of the QMR (quasi-minimal residual) method [14]. FFT routines from the IMSL mathematical library are used to compute the matrix-vector products. The current version uses only one scattering plane for each particle. As discussed for the ZDD code (Section 3.3), the use of multiple scattering planes would increase the efficiency of orientation averaging.

3.2. DDSCAT

The DDSCAT code is based on the DDA, and was implemented by Draine and Flatau [15]. The code is written in Fortran 77, and is highly portable. The current release is version 6.1; complete source code and extensive documentation are freely available.[1] The code is able to automatically generate a number of standard target shapes (e.g. spheres, spheroids, ellipsoids, rectangular solids, tetrahedra, cylinders, hexagonal prisms). In addition, the user has the option of supplying a list of occupied lattice sites to describe any desired target geometry. DDSCAT is capable of treating anisotropic dielectric tensors, and multicomponent targets. The code calculates total scattering and absorption cross sections, as well as user-selected elements of the Mueller scattering matrix in user-selected scattering directions.

The parameters of the scattering problem (target shape, size, orientation, and desired scattering calculations) are read from a parameter file. The user also supplies a separate file with the tabulated dielectric function or refractive index for each material as a function of wavelength; DDSCAT will interpolate to obtain the dielectric function at the wavelengths requested in the parameter file.

DDSCAT uses the GPFA FFT routines [16,17] or FFTW 2.1 routines [26] to compute the necessary matrix-vector products [18]. DDSCAT solves the scattering problem iteratively, using complex conjugate gradient methods. Two options are offered: (1) the preconditioned biConjugate gradient with stabilization method from the parallel iterative methods (PIM) package created by Dias da Cunha and Hopkins; (2) the complex conjugate gradient algorithm of Petravic and Kuo-Petravic [19]. All required routines are supplied with the DDSCAT distribution.

Version 6.1 supports MPI, and can run separate scattering problems (e.g. different target orientations or wavelengths) simultaneously on multi-CPU systems.

[1] http://www.astro.princeton.edu/~draine/.

3.3. ZDD

The code ZDD has been developed at the Astronomical Institute of Kharkov National University in the Ukraine [12,20,21]. Since 1999 Zubko and Shkuratov have worked with the algorithm and code development. The ZDD code has been written in C++. This code uses several ideas that are used in the DDSCAT code, e.g. for the determination of polarizability of dipoles ZDD uses the relationship obtained by Draine and Goodman [9]. Since 2001 the ZDD code uses the fast Fourier transformation as has been suggested by Goodman et al. [18]. There are a few principal distinctions and advantages of the ZDD code as compared to DDSCAT:

(a) *The averaging of particle scattering properties over scattering planes*: The DDA computations can be divided into two stages. The first one is when the field induced on each dipole is calculated. The second one corresponds to the determination of the scattered field in the far zone. Since the first stage consumes a lot of computation time, the ZDD algorithm uses the first stage results to calculate scattering properties in a few different scattering planes (which can also be done in DDSCAT and ADDA). Empirically we found that the use of several scattering planes may considerably improve the averaging over particle orientations. In this work we use four evenly placed scattering planes per one orientation of the wave vector of the incident field, but in principle it is possible to increase the number. For example, we use 100 scattering planes for calculations of averaged scattering properties of very irregular particles, obtaining reliable results much faster than using one scattering plane. In particular, it ensures that the degree of linear polarization will be close to 0% at exact forward and backward directions even when the Monte Carlo averaging over orientations produces fairly poor results.

(b) *Solution of system of linear algebraic equations*: The scheme used in ZDD belongs to the family of conjugate gradient methods together with the BiConjugate Gradient Stabilized method (BCGSM) which is a basic scheme of DDSCAT, but has another formulation [22].

We introduce this scheme here. Let a system of linear algebraic equations be $\mathbf{A}\mathbf{x} = \mathbf{B}$ and let i ($i = 0, 1, 2, \ldots$) be the number of the current iteration step. The residual vector between the current approximation i and the exact solution of the system is denoted with $\mathbf{r}^i$. For an arbitrary initial approximation of the solution $\mathbf{x}^0$, the residual vector $\mathbf{r}^0$ is formulated as $\mathbf{r}^0 = \mathbf{B} - \mathbf{A}\,\mathbf{x}^0$. Set also additional auxiliary vector $\mathbf{s}^i$ and assume that $\mathbf{s}^0$ to be equal to $\mathbf{r}^0$. With these definitions, the computation for the ith iteration step is the following:

$$\begin{aligned}
\alpha_i &= \frac{\mathbf{r}^i \cdot \mathbf{s}^i}{\mathbf{s}^i \cdot (\mathbf{A}\,\mathbf{s}^i)},\\
\mathbf{x}^{i+1} &= \mathbf{x}^i + \alpha_i\,\mathbf{s}^i,\\
\mathbf{r}^{i+1} &= \mathbf{r}^i - \alpha_i\,\mathbf{A}\,\mathbf{s}^i,\\
\beta_i &= \frac{\mathbf{r}^{i+1} \cdot (\mathbf{A}\,\mathbf{s}^i)}{\mathbf{s}^i \cdot (\mathbf{A}\,\mathbf{s}^i)},\\
\mathbf{s}^{i+1} &= \mathbf{r}^{i+1} + \beta_i\,\mathbf{s}^i.
\end{aligned} \tag{10}$$

This scheme uses only one matrix-vector product per iteration as distinct from the DDSCAT code scheme which uses two matrix-vector products per iteration. At the same time, it was found that typically our scheme provides a solution with only a slightly larger number of iteration steps than BCGSM. Thus, our code consumes less computational operations. Additionally, we have found that at critical sets of parameters, for example, with a large value of the size parameter the BCGSM fails while our scheme still works properly.

The scheme with one matrix-vector product per iteration has a certain restriction—it requires that the matrix of coefficients $\mathbf{A}$ should be symmetrical, which means, in practice, that the particle calculated should be homogeneous. Thus, in cases when it is necessary to calculate scattering of light by inhomogeneous objects our code applies BCGSM, just as the DDSCAT code.

(c) *Computation rounding errors*: We have empirically found that the accuracy of numerical presentation of variables plays a significant role in the convergence of the iteration process. For instance, for a sphere placed in a cubic matrix with size of 64 cells at $x_{\mathrm{eq}} = 10$ and $m = 1.6 + 0i$ the number of iterations for incident wave polarized in scattering plane is 541 when each variable has 4 bytes size (in the C programming language it is

called float type) and only 374 when the size is 8 bytes (double type). Unfortunately, the penalty for increasing sizes of all variables is the increased computer memory consumption. That is why we have carried out several experiments to optimize the convergence of the iterative process by increasing sizes of only some variables. It was found that calculations of coefficients α_i and β_i are critical for rounding errors. We used the double type presentation for these two variables in computations while keeping the other variables float type. This simple modification reduced the number of iterations in the experiment mentioned above from 541 to 411. Of course, it is slightly more than 374 steps, but the memory requirements are practically the same as for computation with single accuracy. We use the same trick with the BCGSM calculations.

(d) *Parallelization of calculations*: Our parallelization of DDA computations has some specific features. A few tens of personal computers (PC) of different performance connected with a regular ethernet network are in our disposal. The task is to calculate light scattering of an ensemble of random irregular particles, which implies averaging over orientations and samples. Therefore, our computing parallelization is a distribution of the tasks among the array of personal computers. For a further use of the calculation results, each ZDD process stores results (the number of orientations/samples, values of extinction and absorption cross sections, and phase dependences of all elements of Mueller matrix) in a binary file as a sequence of bytes, and then it is possible to combine results from the PCs without losing the accuracy. This kind of parallelization is quite primitive but it has some gains. Because each computation process is entirely independent of others one can use all the single processor resources in computation (the processor does not wait for results of others and it is not necessary to make data swap between processors). We also can operate with computers of different performance. Computation with independent processors can recover from the possible crashes of operating systems or hardware devices.

(e) *Several minor useful advantages*: The described storing of DDA results in a binary file without losing the accuracy is used also for recovering calculation if the system is crashed. Additionally, the binary file allows us to see the intermediate results while calculations are still continuing. Our ZDD code does not require to be recompiled for each new computation, since it reads values from the command line to obtain the set of task parameters. The names of files with results are not fixed and they can be chosen according to users' wish. The ZDD code uses a few predefined algorithms to produce samples of particles with irregular shape and internal structure. Since the code has been developed only with libraries of ANSI standard of C programming language and only overloading of function has been used from the C++ standard, the ZDD code can be easily transferred to any compiler or platform that supports the C/C++ ANSI standard.

3.4. ADDA

The 'Amsterdam DDA' (ADDA) has been developed for more than 10 years at the University of Amsterdam [23,24]. Its main feature (distinctive from other DDA codes) has always been the capability of running on a cluster of computers (parallelizing a single DDA computation), which allows using a practically unlimited number of dipoles, since ADDA is not limited by the memory of a single computer [24,25]. Recently, the overall performance of the code has been significantly improved, together with some optimizations specially for single-processor mode. Source code and extensive documentation for ADDA are freely available.[2] Version 0.7a was used in this paper, however, current release is 0.75.

Most of ADDA is written in ANSI C, which ensures wide portability. The code is fully operational under Linux and, in sequential mode, under Win32 operating systems. Double precision is used throughout the code; this may improve the convergence of the iterative solver. The parallelization is now performed using message passing interface (MPI). The fast Fourier transform (FFT) is performed either using routines by Temperton [16] or the more advanced package 'fastest Fourier transform in the west' (FFTW) [26]. The latter is generally considerably faster but requires separate package installation. FFTW package was use in this comparison.

ADDA has several options implemented for dipole polarizabilities: Clausius–Mossotti (CM) [27], radiative reaction correction (RR) [28], lattice dispersion relation (LDR) [9], and corrected LDR (CLDR) [29]. In this paper the most common LDR (as in DDSCAT) was used.

[2] http://www.science.uva.nl/research/scs/Software/adda/.

ADDA includes four Krylov-subspace iterative methods [30]: conjugate gradient applied to normalized equation with minimization of residual norm (CGNR) [31], Bi-CG STABilized (Bi-CGSTAB) [31], Bi-CG [14], and QMR [14]. The last two iterative methods employ the complex-symmetric property of the DDA matrix to decrease twice the calculation time [14]. The code employs DDA equations in the following form:

$$\overline{\beta}_i \mathbf{E}_i^{inc} = \mathbf{x}_i - \sum_{j \neq i} \overline{\beta}_i \overline{\mathbf{G}}_{ij} \overline{\beta}_j \mathbf{x}_j, \tag{11}$$

where $\overline{\beta}_i$ is any matrix square root of the dipole polarizability $\overline{\alpha}_i : \overline{\beta}_i = \sqrt{\overline{\alpha}_i}$ (it always exists for diagonal polarizabilities). $\mathbf{E}_i^{inc}$ is incident electric field, and $\overline{\mathbf{G}}_{ij}$ is free-space Green's tensor (complex symmetric). The unknown vector $\mathbf{x}_i$ is connected to the dipole polarizations $\mathbf{P}_i$ by $\mathbf{P}_i = \overline{\beta}_i \mathbf{x}_i$. Eq. (11) is equivalent to use of Jacobi-preconditioning [31] together with keeping the interaction matrix complex-symmetric (for any distribution of refractive index inside the scatterer and for any of the supported polarization prescriptions). The default stopping criterion of the iterative method in ADDA is the relative error of the residual, which must be smaller than 10^{-5}. In our experience QMR is generally the fastest of the supported methods, which was also suggested by Rahola [32], hence it was used in this paper.

There are several factors that allows ADDA's performance to compare favorably with other codes. They include the use of FFTW 3 package that adapts to any particular hardware. Moreover, ADDA does not perform complete 3D FFT transform in one run, but decomposes it into a set of 1D transforms with data transposition in between. It allows to employ the fact that input data for the forward transform contain a lot of zeros and only part of the output data of the backward transform are needed. QMR iterative solver seems to be the most suitable for the targets used in this paper. Moreover, iterative solvers in ADDA are optimized for the specific task, which allows removal of some unnecessary computations at the cost of code modularity. Dynamic memory allocation and optimized data structure allow all computations except FFT to be performed only for the real (non-void) dipoles and not for the whole computational box. This also decreases ADDA's memory consumption. Finally, symmetry of the interaction matrix is used to decrease storage requirement of its Fourier transform.

Orientation averaging is done over three Euler angles (α, β, γ). Rotating over α is equivalent to rotating the scattering plane without changing the orientation of the scatterer relative to the incident radiation. Therefore, averaging over this orientation angle is done with a single computation of internal fields; additional computation time for each scattering plane is comparably small. Averaging over the other two Euler angles is done by independent DDA simulations. The averaging itself is performed using Romberg integration [30], which may be used in adaptive regime (automatically simulating necessary number of different orientations to reach prescribed accuracy) but limits the possible number of values for each orientation angle to be $2^n + 1$ (n is any integer). In most situations starting and ending values of both α and γ are equivalent and ADDA accounts for it to slightly decrease simulation time. Moreover, symmetries of the scatterer may be used to decrease the intervals of Euler angles, over which to average, and hence accelerate the calculation, however, it has not been used in this paper.

4. Comparison between the codes

We have compared the four DDA-based light scattering codes to each other and against exact solutions from codes based on the T-matrix or Mie approach. The comparison has been done with five different scattering geometries and with two refractive indices. Four of the geometries were calculated in random orientation. Differences from exact results are reported, as well as CPU times and computer memory requirements.

There are several different areas that one needs to take into account when programming a good DDA code. These involve both theoretical and practical issues, such as:

- The user interface (if any) for the code (programming problem).
- The choice of polarizabilities for the dipoles (physical and theoretical problem).
- The choice of solver for the system of linear equations (applied mathematics problem).
- The programming of the solver (programming problem).

From the user's point of view, all of these should be done well before the DDA code is useful.

4.1. Geometries and parameters

We have used the following five geometries: sphere, spheroid, cylinder and two clusters of uniform spheres, a 4-sphere and a 50-sphere cluster. All geometries have the same size (volume), and the size parameter $x_{eq} = 2\pi r_{eq}/\lambda$ is 5.1, where r_{eq} is the equal-volume-sphere radius and λ is the wavelength.

The geometries must be represented by rectangular array of dipoles for the DDA. The number of dipoles affects the accuracy of the result, but in the same time it affects the CPU time and memory consumption. Draine and Flatau give a rule of thumb for the dipole size in the user manual of DDSCAT [33]. They state that the dipole size must be small compared to:

- Any structural length in the target geometry.
- The wavelength λ. This criteria is satisfied if

$$|m|\frac{2\pi}{\lambda}d \leqslant \frac{1}{2}, \tag{12}$$

where m is the complex refractive index of material and d is the dipole size or the distance between two adjacent dipoles.

We have decided to use two refractive indices throughout the comparison, index m_s is $1.6 + 0.001i$ (silicate) and index m_i is $1.313 + 0i$ (ice). For m_s, d must be smaller than 0.3125 in size parameter units to satisfy Eq. (12). We have used a value $d = 0.3$, varying it just a little to ensure that the volume stays the same for all the geometries. SIRRI has similar internal check for d: the number of cells in one wavelength inside the material must be over 10, which also agrees with the choice of $d = 0.3$.

With this dipole size and the size parameter each of the geometries requires about 21 000 dipoles for material. The memory requirements of DDA are determined by the size of the rectangular grid that contains all the dipoles. For a sphere, these ~21 000 dipoles fit to a cubic grid with side length of 35 dipoles. The most porous shape, cluster of 50 spheres, needs a rectangular grid of $57 \times 59 \times 62$ dipoles. The dipole presentations of the shapes are shown in Fig. 1.

Sphere is a rotationally symmetric shape, thus no orientation averaging is needed for the DDA computations of its scattering, although it might improve the accuracy by decreasing the errors due to the discrete representation of the shape. Sphere was computed in single orientation. For the other shapes, results must be averaged over orientations if we want to compare the results to the random orientation T-matrix results. We have computed results for different number of orientation averages, but we report here only the results that use 1024 orientations (actually, for technical reasons DDSCAT uses 1089 and ADDA 1152 orientations). The tolerance for DDA convergence is 10^{-5} for all the codes.

The calculation over random orientation of the scatterer is implemented in different ways in these four codes. The three Euler rotation angles (α, β, γ) can be sampled either randomly or systematically. Furthermore, as mentioned in Sections 3.3 and 3.4, the rotation over α can be replaced by using corresponding scattering planes which will fasten up the calculations. From the DDA codes discussed here, SIRRI uses full random sampling over all the three angles. ZDD uses also random sampling, but in addition it calculates scattering with four scattering planes per orientation. Thus, the number of 1024 orientations is reached with 256 rotations and with four scattering planes per orientation. DDSCAT and ADDA use systematic sampling of Euler angles. We have chosen to use 11 scattering planes and 9 (β) and 11 (γ) orientation angles with DDSCAT, producing a total of 1089 orientations. With ADDA the choice of the number of angles is more limited, since all the values must be powers of two (plus one in some cases). We have chosen to use 16 scattering planes with ADDA and 9 (β) and 8 (γ) orientation angles, producing a total of 1152 orientations.

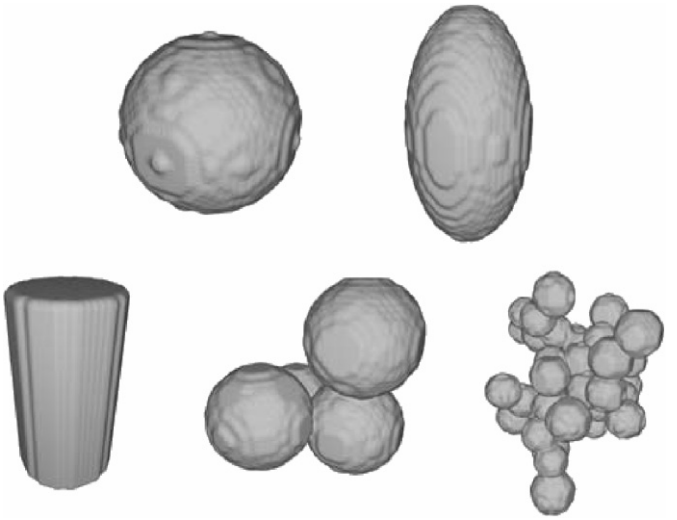

Fig. 1. The geometries used in the comparison. From the top left corner, a sphere and a spheroid. The second line from left, a cylinder, a 4-sphere and a 50-sphere cluster. Both the spheroid and the cylinder have an aspect ratio (diameter divided by height) of $\frac{1}{2}$. These visualizations are drawn from the discretized dipole representations of the shapes. One can see how the shapes deviate a bit from their actual shape.

4.2. Exact methods

We have chosen the geometries for this study so that the scattering can also be computed with either the Mie (sphere) or the T-matrix method. We will consider the T-matrix and Mie results to be practically exact, and compare all the DDA results against these.

The different exact codes used here are:

- Mie code from Lompado for sphere. The code is written for Mathematica.
- T-matrix code from Mishchenko and Travis for cylinder and spheroid [6]. The code is written in Fortran 77 language.
- Cluster T-matrix code from Mackowski and Mishchenko for sphere clusters [7]. The code is written in Fortran 77 language.

4.3. Results

4.3.1. Accuracy

The results from DDA codes are compared to the results provided by the exact methods. All the codes produce the full Mueller matrix $\mathbf{M}$ for a given set of the scattering angles θ. We use a range for θ from 0° to 180° with one degree steps. We will report here only the scattered intensity I (the element M_{11} in $\mathbf{M}$) and the linear polarization ratio P ($-100\%\ M_{21}/M_{11}$).

The magnitudes of values for I vary a lot, and I is often plotted in a logarithmic scale. Therefore we feel that the absolute error against exact solution is not as informative as the relative error $\varepsilon_I(\theta) = 100\%\ (I_{\mathrm{DDA}}(\theta) - I_{\mathrm{exact}}(\theta))/I_{\mathrm{exact}}(\theta)$. For the polarization ratio P, the absolute error $\varepsilon_P(\theta) = P_{\mathrm{DDA}}(\theta) - P_{\mathrm{exact}}(\theta)$ is more suitable. These errors, together with the exact solutions, are presented in Figs. 2–6 for all the geometries and for both the refractive indices.

To get an overview on the average accuracy of the DDA codes, we report also the mean absolute errors (MAE) for the codes in Tables 1 and 2. We prefer $MAE = 1/181 \sum_{\theta=0^\circ}^{180^\circ} |\varepsilon(\theta)|$ over the commonly used root mean square error because MAE is more intuitive in describing the average error if no further statistical analysis is needed.

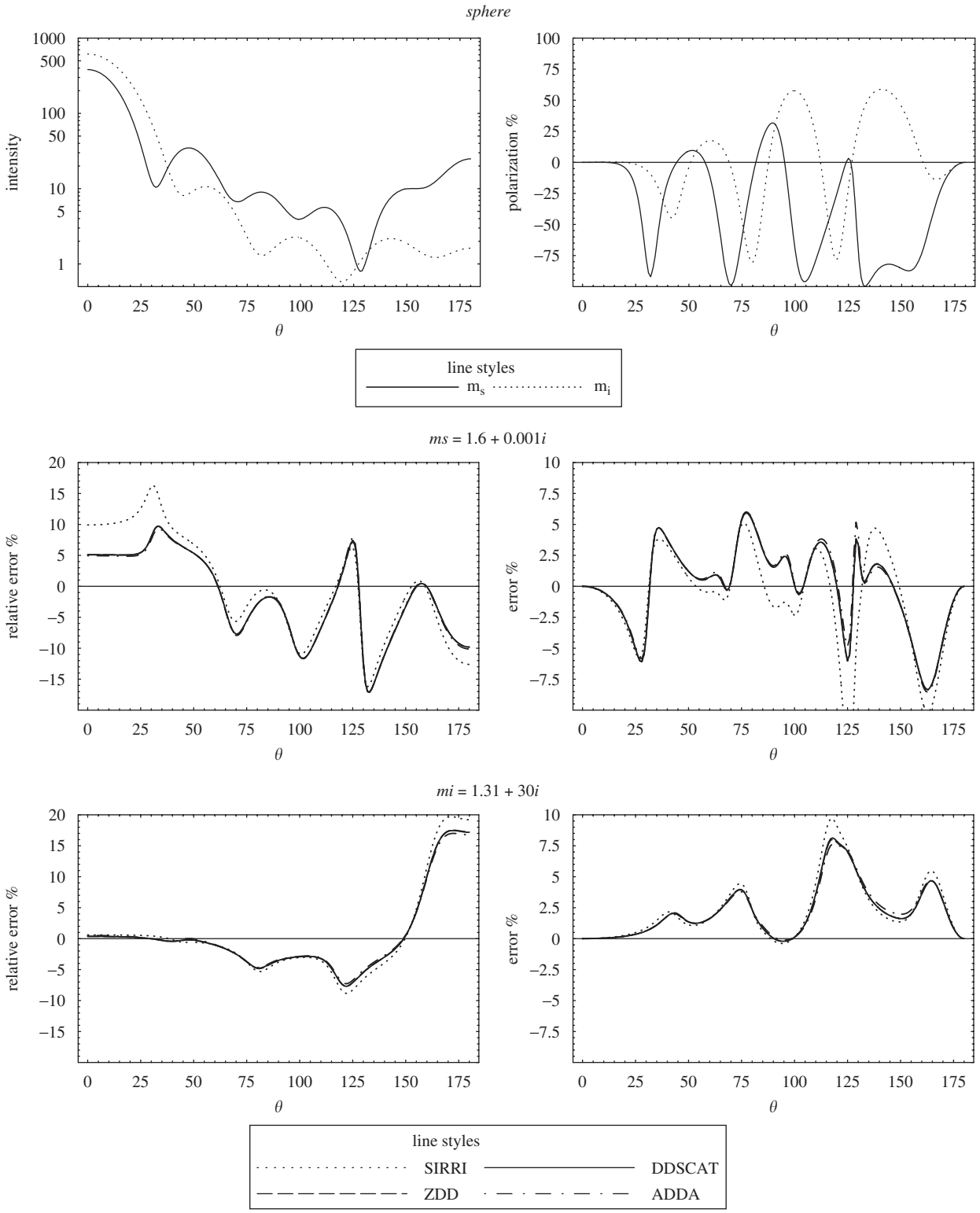

Fig. 2. Comparison results for sphere. The exact solutions for I and P are shown on the first row. The errors of DDA codes are shown on the second row for m_s and on the third row for m_i. Intensity errors are on the left and polarization errors on the right. Note that the x and y scales in the figures are the same in all the Figs. 2–6.

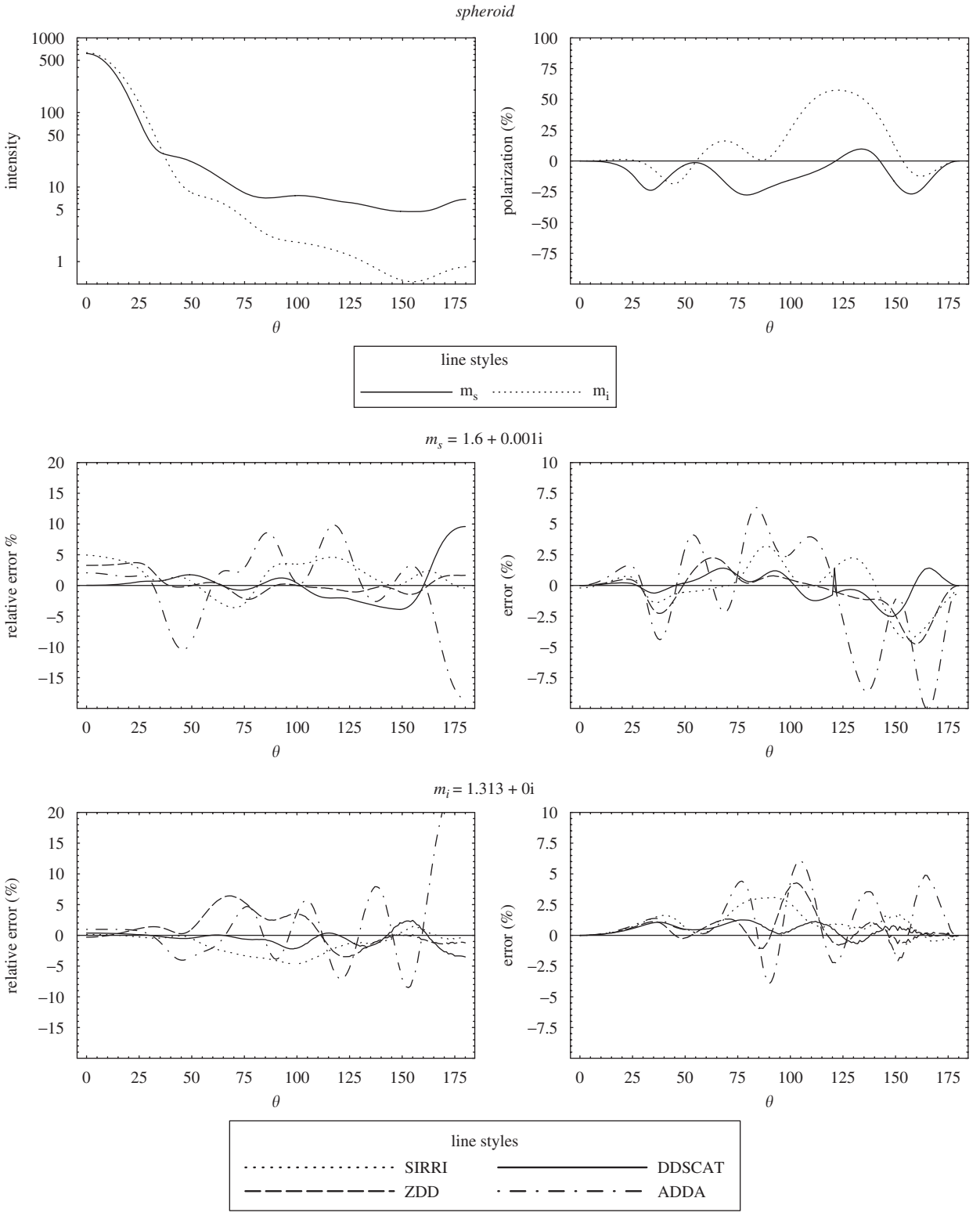

Fig. 3. Comparison results for randomly oriented spheroid.

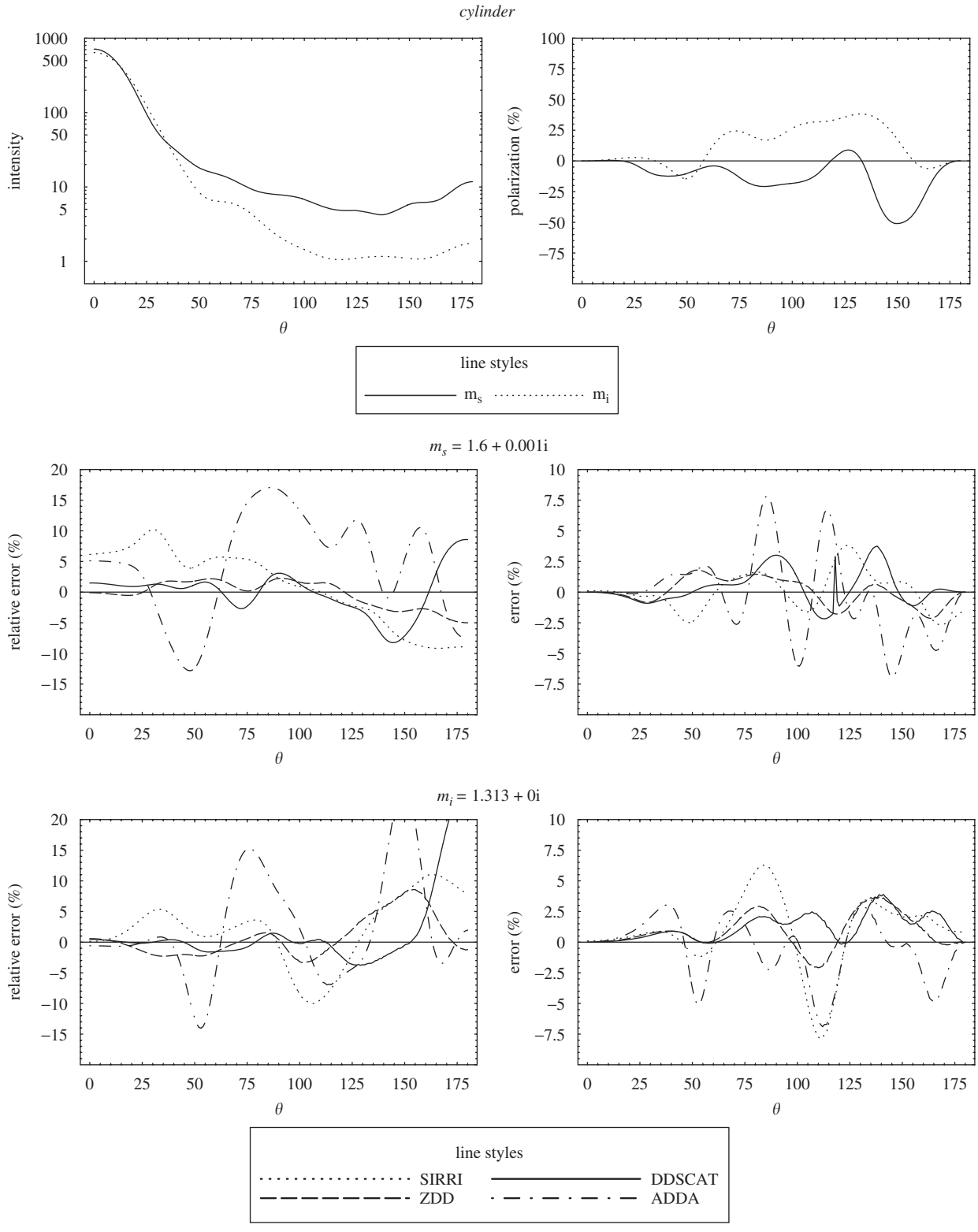

Fig. 4. Comparison results for randomly oriented cylinder.

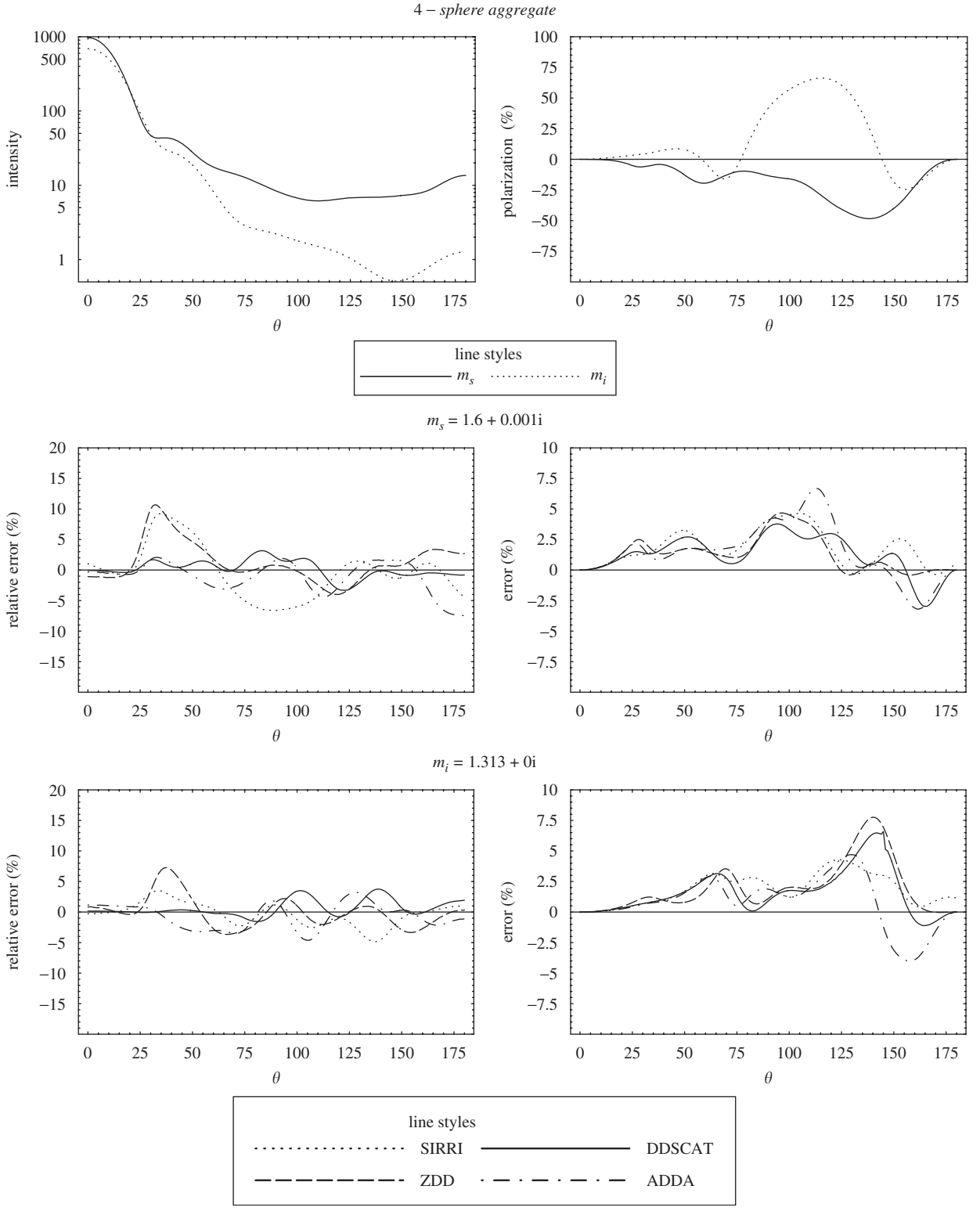

Fig. 5. Comparison results for randomly oriented 4-sphere aggregate.

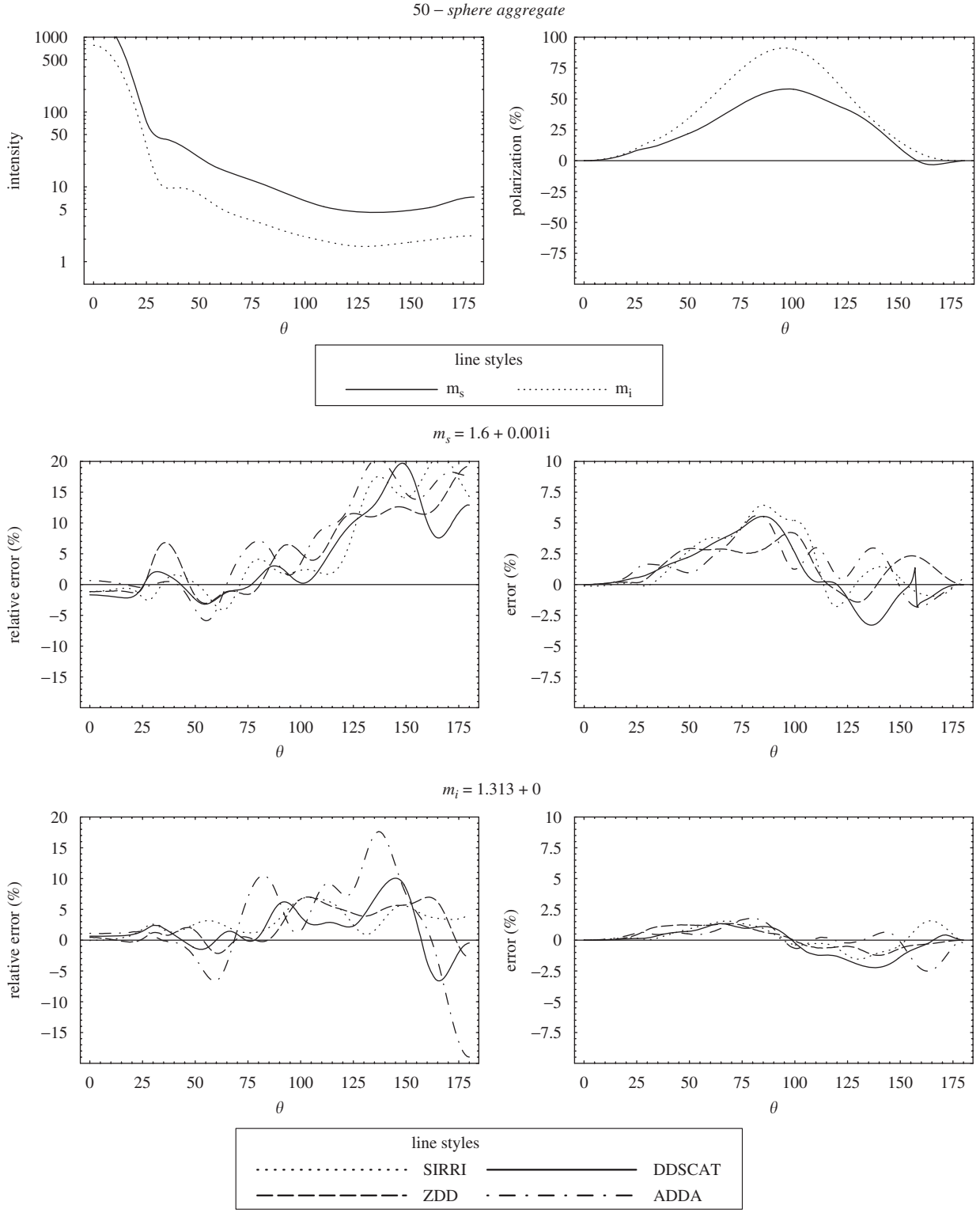

Fig. 6. Comparison results for randomly oriented 50-sphere aggregate.

Overall it seems that the typical relative error for I for DDA codes is between 2% and 6% for silicate and ice type refractive indices and for particles with size $x_{eq} \approx 5$. For P, the typical absolute error is roughly from 1% to 3%. Both the errors tend to be larger for larger refractive index. The overall accuracy of the solution is best with DDSCAT and almost as good with ZDD, and not so good with SIRRI and ADDA.

If we compare the different approaches to the calculation of randomly oriented scatterers we can notice that both the systematic sampling and the random sampling of rotation angles can give accurate results. The most accurate code, DDSCAT, uses systematic sampling. ZDD, which is almost as accurate, uses random sampling with the addition of four symmetric scattering planes. However, the random sampling in SIRRI does not perform that well. ADDA has good accuracy in the single orientation case with sphere, but the orientation averaging scheme might need some improvements since the accuracy is not that good with geometries in random orientation. The choice of the number of orientation angles is not that flexible with ADDA, and there is a trade-off between speed (more scattering planes) and accuracy (more orientation angles). Our choice of 16 scattering planes and $9 * 8$ orientations with ADDA gives quite good speed, faster than other codes by a factor of 2.3 to 16, but other choices might give somewhat improved accuracy.

If we take a closer look in the structure of errors in Figs. 2– 6 we can see that the errors are mainly randomly distributed. For instance, if we take the mean of errors (including the sign), the mean errors in most cases are randomly either positive or negative. Exceptions are the mean errors in P with ZDD which are positive in eight cases out of ten, and the mean errors in I for the 50-sphere cluster which are positive for all the codes.

Table 1
Mean absolute errors (*MAE*) for intensity (I) and linear polarization ratio (P) for all the DDA codes with five geometries and two refractive indices

		SIRRI		DDSCAT		ZDD		ADDA	
		m_s	m_i	m_s	m_i	m_s	m_i	m_s	m_i
Sphere	I	*7.027*	*4.602*	5.944	4.019	5.938	4.030	**5.921**	**3.896**
	P	*2.952*	*2.371*	**2.499**	2.192	2.507	**2.183**	2.522	2.206
Spheroid	I	2.362	1.562	1.918	**0.905**	**1.218**	1.956	*4.590*	*4.823*
	P	1.314	1.109	**0.727**	**0.527**	1.151	0.798	*3.263*	*1.818*
Cylinder	I	5.373	4.743	2.627	2.926	**1.758**	**2.410**	*8.090*	*6.436*
	P	1.210	*2.216*	0.999	**1.204**	**0.869**	1.209	*2.370*	1.897
4-Sphere aggregate	I	*3.203*	1.473	**1.099**	**0.964**	2.438	*1.860*	1.995	1.772
	P	1.830	1.882	1.676	**1.761**	**1.422**	*2.007*	*2.079*	1.780
50-Sphere aggregate	I	6.473	2.987	**5.573**	2.925	6.458	**2.878**	*7.811*	*6.017*
	P	*2.026*	0.711	1.866	*0.834*	**1.660**	**0.656**	1.826	0.690

Sphere is calculated in single orientation, other geometries are randomly oriented. Errors for intensity are relative to the exact solution while errors for polarization are absolute errors and both are expressed in percentages. For every geometry, refractive index and either I or P, the smallest error is printed in bold font and the largest error in italics.

Table 2
Averages of *MAE* over all the geometries from Table 1

	SIRRI		DDSCAT		ZDD		ADDA	
	m_s	m_i	m_s	m_i	m_s	m_i	m_s	m_i
I	4.888	3.073	**3.432**	**2.348**	3.562	2.627	*5.681*	*4.589*
P	1.866	1.658	1.553	**1.304**	**1.522**	1.371	*2.412*	*1.678*

The smallest error for every category is printed in bold font and the largest error in italics.

More systematic trends can be seen if we divide the θ range in forward scattering ($\theta \in [0^\circ, 90^\circ]$) and in backward scattering ($\theta \in [90^\circ, 180^\circ]$) regions. In all 80 combinations of the four codes, five geometries, two refractive indices and I and P, the errors in backward scattering are larger in 63 cases. This definitely implies to a systematic behavior in the error structure of the DDA codes.

The integrated scattering quantities, e.g. $Q_{\text{ext}}, Q_{\text{sca}}, Q_{\text{abs}}$ and $\langle \cos\theta \rangle$ are needed in many applications. The integration over θ usually improves the accuracy. The average accuracy of the codes over the geometries for Q_{sca} and $\langle \cos\theta \rangle$ are presented in Table 3. DDSCAT is the best for these integrated quantities, followed by ZDD. The relative errors for Q_{sca} and the absolute errors for $\langle \cos\theta \rangle$ are typically under 1% for most of the codes.

4.3.2. Speed

Besides the accuracy, the CPU time and the memory requirements of a DDA code are very important factors. Both of them currently limit the possibilities of DDA approaches in scattering studies. DDA is very flexible because it can handle all kinds of scattering geometries, but the size parameter of the scatterer is limited. PC computers nowadays can have few gigabytes of operating memory and for a scatterer that requires few gigabytes of memory, the CPU time needed for a single orientation can be in the order of one CPU day. Scattering problems larger than this are not very practical to solve with DDA. With the DDA codes studied here, this size limit is reached with a rectangular grid of dipoles with side length of about 150 to 200 dipoles. This means that using the condition on Eq. (12), the side length of the rectangular volume that encloses the scatterer is about 50 to 70 in size parameter units, and the volume-equivalent sphere radius is about 30 to 45 in size parameter units.

Some examples of the CPU times and memory consumptions of the DDA codes in our study are presented in Tables 4 and 5. The codes are run with the IBMSC computer at the Finnish Center for Scientific Computing (CSC). The IBMSC is an IBM eServer Cluster 1600 supercomputer consisting of IBM p690 nodes that use Power4 processors running at 1.1 GHz frequency. The computer is designed for efficient parallel computing and if serial programs that use only one processor are executed, the performance is comparable to a modern PC.

In the IBMSC environment, the latest version of ADDA seems to be by far the fastest. The SIRRI code is also quite fast in computing single orientation. When averaging over several orientations, SIRRI will not perform that well. As mentioned in Sections 3.3 and 3.4, the DDA averaging over different orientations can be divided in two stages, and computing efforts can be saved computing results for multiple scattering planes for every first stage (internal field calculation) result. All the other codes except SIRRI use this scheme. The extra computational task for computing more than one scattering plane per internal field is minimal. The results for the spheroid in Table 4 are computed over 80 orientations. For DDSCAT, ZDD and ADDA, 20 orientations where the internal field is computed are used and four scattering planes are computed per every internal field.

It should be kept in mind that the efficiency of a code can depend on the computing environment (processor type, compiler and operating system). For example, DDSCAT is faster than ZDD in IBMSC, but with a x86 Windows PC ZDD compiled with Watcom C++ is faster than DDSCAT compiled with Absoft Pro Fortran (Table 5).

Table 3
Averages of MAE errors over all the geometries from Table 1 for integrated quantities Q_{sca} and $\langle \cos\theta \rangle$

	SIRRI		DDSCAT		ZDD		ADDA	
	m_{s}	m_{i}	m_{s}	m_{i}	m_{s}	m_{i}	m_{s}	m_{i}
Q_{sca}	*2.325*	0.5103	**0.7307**	**0.2603**	1.210	0.5397	1.207	*0.5535*
$\langle \cos\theta \rangle$	*1.266*	0.1699	0.8304	**0.1383**	**0.7968**	0.1562	0.9164	*0.2559*

Errors for Q_{sca} are relative to the exact solution. The $\langle \cos\theta \rangle$ is already a ratio of integrated intensities with and without the $\cos\theta$ term and therefore absolute error is used. Both the errors are expressed in percentages. The smallest error for every category is printed in bold font and the largest error in italics.

Table 4
CPU times and memory consumptions for the DDA codes for two example cases

			CPU-time per orientation (s)	Memory per processor (Mbyte)
Cube	m_s	SIRRI	45	54.1
1 orientation		DDSCAT	51	22.5
21 952 dipoles		ZDD	273	21.6
		ADDA	18	15.3
	m_i	SIRRI	19	54.1
		DDSCAT	37	22.5
		ZDD	85	21.6
		ADDA	9	15.3
Spheroid	m_s	SIRRI	137	74.8
80 orientations		DDSCAT	13	32.3
20 775 dipoles		ZDD	**	**
		ADDA	11	21.6
	m_i	SIRRI	41	74.8
		DDSCAT	10	32.3
		ZDD	13	**
		ADDA	5	21.6

**indicates that the value has not been measured.

Table 5
CPU-times for cube with sidelength of 64 dipoles and with $m = m_i$ in single orientation for DDSCAT and ZDD compiled and executed in either IBMSC or PC environment

		CPU-time
PC	DDSCAT	9 min 33 s
	ZDD	8 min 43 s
IBMSC	DDSCAT	8 min 42 s
	ZDD	11 min 17 s

5. Discussion

All the codes compared here have some good qualities and also some drawbacks. For example, DDSCAT and ZDD produce accurate results, ADDA is fast and SIRRI is very flexible to use allowing e.g. user-specified list of orientation angles. All the codes can run parallel using more than one processor and ADDA can use several processors even for one orientation. DDSCAT and ADDA are freely available. On the other hand, DDSCAT cannot use dynamical memory allocation, DDSCAT and ADDA cannot handle averaging over user-specified orientation angle distribution, SIRRI has poor accuracy with polarization in the backscattering domain etc.

Pros and cons of the codes

SIRRI

+ Fast code for single orientation.
+ Fortran 90 code. Has dynamical memory allocation and reads the name for the parameter file from the command line so it needs to be compiled only once.
+ The format for parameter file is flexible.
+ Can calculate scattering for anisotropic material.

+ User can specify a list of orientation angles.
+ Updates the result file while calculating and it is possible to add more averages later.

− Not freely available.
− Results are among the two most inaccurate codes.
− Polarization in $\theta = 180^\circ$ is not zero.
− Slow on random orientation.
− Most memory-demanding.

DDSCAT

+ The most accurate code.
+ The second fastest code in supercomputer environment when calculating in random orientation.
+ Free.
+ Full source code available, including the parallelized MPI version.

− Code is written in Fortran 77 and it needs recompiling for different sized geometries.
− Code uses fixed name for its parameter file, so simultaneous calculations must be run in separate directories.
− Cannot handle user-specified distribution of orientation angles unless the angles combine all the elements with each other in the three user-specified rotation angle lists. This might be fixed in the future versions.
− If system crashes before the program has stopped all the calculations need to be repeated. If user needs more averages the results must be combined manually.

ZDD

+ Very accurate, almost as good as DDSCAT.
+ Fast on x86 processors.
+ Dynamical memory allocation and parameters from command line.
+ Can recover from system crash and combine more orientations with the old results.
+ Can be run 'parallel' without any message passing by combining the results from different computers afterwards.

− Slowest on the supercomputer environment with single orientation.
− The code is not publicly available.

ADDA

+ The fastest and least memory consuming code.
+ Can be run in parallel using MPI, parallelizing a single DDA computation, thus allowing simulation of very large particles.
+ Free. Full source code and extensive documentation are available.
+ Dynamical memory allocation and reading of parameters from the command line. Input–output interface is designed for multiple parallel simulations.
+ Checkpoint system available to recover from system error or split large calculations.
+ It is possible to simulate light scattering by a tightly focused Gaussian beam.
+ Orientation averaging can be performed in an adaptive regime, i.e. to reach prescribed accuracy. It comes at the cost of loss of flexibility to specify a set of orientation angles.

- Accuracy of the orientation averaging scheme is being improved.
- Anisotropic materials currently cannot be used.
- Only few predefined shapes are currently available, however, dipole configuration can be read from a file that has flexible format.

Acknowledgments

A. Penttilä is thankful for the financial support of the Finnish Cultural Foundation and the Magnus Ehrnrooth foundation and E. Zubko for the financial support of the Academy of Finland. The computations presented in this article have been made with CSC's computing environment. CSC is the Finnish IT center for science and is owned by the Ministry of Education.

References

[1] Mishchenko MI, Travis LD, Lacis AA. Multiple scattering of light by particles. Radiative transfer and coherent backscattering. New York: Cambridge university press; 2006.
[2] Chandrasekhar S. Radiative transfer. New York: Dover; 1960.
[3] Sobolev VV. Light scattering in planetary atmospheres. Oxford: Pergamon Press; 1975.
[4] van de Hulst HC. Multiple light scattering. New York: Academic Press; 1980.
[5] Muinonen K, Zubko E. Optimizing the discrete-dipole approximation for sequences of scatterers with identical shapes but differing sizes or refractive indices. JQSRT 2006;100:288–94.
[6] Mishchenko MI, Travis LD. Capabilities and limitations of a current FORTRAN implementation of the T-matrix method for randomly oriented, rotationally symmetric scatterers. JQSRT 1998;60:309–24.
[7] Mackowski DW, Mishchenko MI. Calculation of the T-matrix and the scattering matrix for ensembles of spheres. J Opt Soc Am A 1996;13:2266–78.
[8] Lumme K, Rahola J. Light scattering by porous dust particles in the discrete-dipole approximation. Astrophys J 1994;425: 653–67.
[9] Draine BT, Goodman J. Beyond Clausius–Mossotti: wave propagation on a polarizable point lattice and the discrete-dipole approximation. Astrophys J 1993;405:685–97.
[10] Lakhtakia A, Mulholland GW. On two numerical techniques for light scattering by dielectric agglomerated structures. J Res Natl Inst Stand Technol 1993;98(6):699–716.
[11] Rahola J. Efficient solution of dense systems of linear equations in electromagnetic scattering calculations. PhD thesis, Helsinki University of Technology; 1996.
[12] Zubko ES, Shkuratov YuG, Hart M, Eversole J, Videen G. Backscattering and negative polarization of agglomerate particles. Opt Lett 2003;28(17):1504–6.
[13] Lumme K, Rahola J. Comparison of light scattering by stochastically rough spheres, best-fit spheroids and spheres. JQSRT 1998;60:439–50.
[14] Freund RW. Conjugate gradient-type methods for linear-systems with complex symmetrical coefficient matrices. SIAM J Sci Stat Comp 1992;13:425–48.
[15] Draine BT, Flatau PJ. The discrete dipole approximation for scattering calculations. J Opt Soc Am A 1994;11:1491–9.
[16] Temperton C. Self-sorting mixed-radix fast Fourier transforms. J Comp Phys 1983;52:1–23.
[17] Temperton C. A generalized prime factor FFT algorithm for any $N = 2^p 3^q 5^r$. SIAM J Sci Stat Comp 1992;13:676–86.
[18] Goodman JJ, Draine BT, Flatau PJ. Application of fast-Fourier-transform techniques to the discrete-dipole approximation. Opt Lett 1991;16:1198–200.
[19] Petravic M, Kuo-Petravic G. An ILUCG algorithm which minimizes the Euclidean norm. J Comp Phys 1979;32:263.
[20] Zubko ES, Kreslavskii MA, Shkuratov YuG. The role of scatterers comparable to the wavelength in forming negative polarization of light. Solar Sys Res 1999;33:296–301.
[21] Zubko E, Petrov D, Shkuratov Y, Videen G. Discrete dipole approximation simulations of scattering by particles with hierarchical structure. Appl Opt 2005;44:6479–85.
[22] Voevodin VV. Matrices and calculations. Moscow: Nauka; 1984 (in Russian).
[23] Hoekstra AG, Sloot PMA. Coupled dipole simulations of elastic light scattering on parallel systems. Int J Mod Phys C 1995;6: 663–79.
[24] Hoekstra AG, Grimminck MD, Sloot PMA. Large scale simulations of elastic light scattering by a fast discrete dipole approximation. Int J Mod Phys C 1998;9:87–102.
[25] Yurkin MA, Semyanov KA, Tarasov PA, Chernyshev AV, Hoekstra AG, Maltsev VP. Experimental and theoretical study of light scattering by individual mature red blood cells with scanning flow cytometry and discrete dipole approximation. Appl Opt 2005;44:5249–56.
[26] Frigo M, Johnson SG. The design and implementation of FFTW3. IEEE Proc 2005;93:216–31.
[27] Purcell EM, Pennypacker CR. Scattering and adsorption of light by nonspherical dielectric grains. Astrophys J 1973;186:705–14.

[28] Draine BT. The discrete-dipole approximation and its application to interstellar graphite grains. Astrophys J 1988;333:848–72.
[29] Gutkowicz-Krusin D, Draine BT. Propagation of electromagnetic waves on a rectangular lattice of polarizable points, 2004 ⟨http://xxx.arxiv.org/abs/astro-ph/0403082⟩.
[30] Press WH, Flannery BP, Teukolsky SA, Vetterling WT. Numerical recipes in C. The art of scientific computing. Cambridge: Cambridge University Press; 1990.
[31] Barrett R, Berry M, Chan TF, Demmel J, Donato J, Dongarra J, et al. Templates for the solution of linear systems: building blocks for iterative methods. Philadelphia, PA: SIAM; 1994.
[32] Rahola J. Solution of dense systems of linear equations in the discrete-dipole approximation. SIAM J Sci Comp 1996;17:78–89.
[33] Draine BT, Flatau PJ. User guide for the discrete dipole approximation code DDSCAT (Version 6.1), 2004 ⟨http://arxiv.org/abs/astro-ph/0409262⟩.