Comparison between discrete dipole and exact techniques

Antti Penttilä,[1] Evgenij Zubko,[1,2] Kari Lumme,[1] Karri Muinonen,[1] Maxim Yurkin,[3,4]
Yurij Shkuratov,[2] and Alfons Hoekstra[3]

[1]*Observatory, University of Helsinki, P.O. box 14, 00014 University of Helsinki, Finland*
[2]*Astronomical Institute of Kharkov National University, 35 Sumskaya Street, Kharkov, Ukraine, 61022*
[3]*Faculty of Science, Section Computational Science of the University of Amsterdam, Kruislaan 403, 1098 SJ, Amsterdam, The Netherlands*
[4]*Institute of Chemical Kinetics and Combustion, Siberian Branch of the Russian Academy of Sciences, Institutskaya Str. 3, 630090, Novosibirsk, Russia*
e-mail: Antti.I.Penttila@helsinki.fi

Abstract

Four computer codes based on the Discrete Dipole Approximation are compared against each other and against exact solutions for five different scattering geometries. The interest is in the accuracy of intensity and linear polarization values the codes produce and in the computational time needed.

1 Introduction

Most of the existing light scattering programs can be divided into three physically different categories: ray-optics (RO), geometrical optics (GO) and wave optics (WO) approaches. A classical example of the ray-optics treatment is the Radiative Transfer Equation (RTE), see, e.g., van de Hulst [1]. The GO methods are basically the same as the RO complemented only by the diffraction correction close to the forward scattering. Finally, the WO methods can be derived more or less directly from the Maxwell's equations and are, in principle, exact. As a qualitative recipe it can be stated that the RO methods can be applied to sparsely packed geometries where the packing density is less than about 10%. GO approach suits for large solid particles and in all the other cases the exact WO methods are a must. Our long standing interest has been to analyze light scattering by planetary regoliths and products from the paper industry. In these applications the WO methods are the only possible ones because the packing density in both these cases certainly exceeds 10%.

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 major drawback in being an approximation although if infinite CPU time would be possible the results should become exact. The other smaller drawback comes from the fact that if orientation averages are needed then the computationally demanding linear equations must be solved repeatedly.

2 Comparison between the codes

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. For this we naturally need codes which can handle some geometries in a numerically exact manner. A fairly obvious choice is to use Mie code, the T-matrix [2] code and the Cluster T-matrix [3] code. We were able to collect four different DDA versions to do this comparison. We have compared the codes with five different scattering geometries and with two refractive indices. Differences from exact result are reported, as well as CPU-times and computer memory requirements. The DDA codes are *sirri* from Rahola, *ddscat* from Draine and Flatau, *zdd* from Zubko and *adda* from the Amsterdam University group.

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 $x_{\mathrm{eq}} = 2\pi r_{\mathrm{eq}}/\lambda$ 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 codes. 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* [4]. They state that the dipole size must be small compared to (*i*) any structural length in the target geometry, and (*ii*) the wavelength λ. The second criteria is satisfied if

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

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. (1). We have used a value $d \approx 0.3$, varying it just a little to ensure that the volume stays the same for all the geometries. With this dipole size and the size parameter each of the geometries requires about 21 000 dipoles. The memory requirements of DDA is determined by the size of the rectangular grid that contains all the dipoles for material. 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 57x59x62 dipoles. The dipole presentation of all the geometries is shown in Fig. 1.

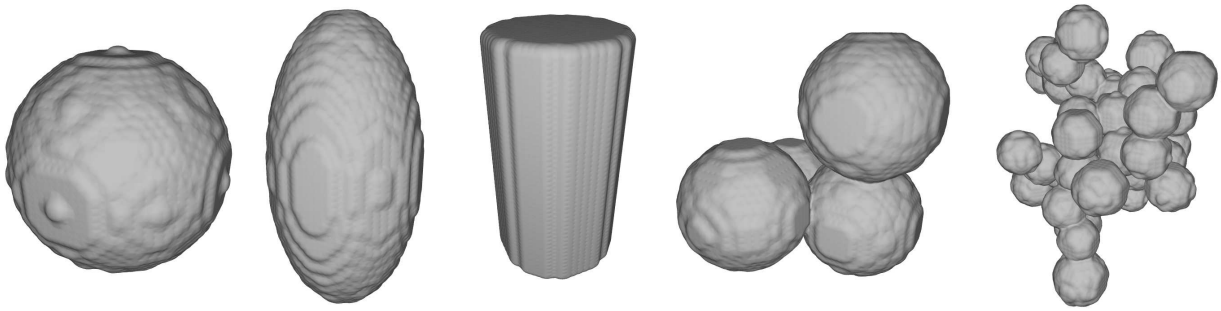

Figure 1: The geometries used in the comparison. From the left, sphere, spheroid, cylinder, a 4-sphere and a 50-sphere cluster. Both the spheroid and the cylinder have an aspect ratio (diameter divided by height) of 1/2.

Sphere is rotationally symmetric shape, thus only one orientation is needed for the DDA calculations of its scattering. 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 calculated 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.

3 Results

The results from DDA codes are compared to the results provided by the exact methods. All 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 pure error against exact solution is not as informative as the relative error $\epsilon_I(\theta) = 100\% \, (I_{\mathrm{DDA}}(\theta) - I_{\mathrm{exact}}(\theta))/I_{\mathrm{exact}}(\theta)$. For the polarization ratio P, the pure error $\epsilon_P(\theta) = P_{\mathrm{DDA}}(\theta) - P_{\mathrm{exact}}(\theta)$ is more suitable. Due to the limited length of this abstract the figures showing the angular behavior of these errors will not be presented here, but they will be available in our poster at the conference. To get an overview on the average accuracy of the DDA codes, we report here the mean absolute errors (MAE) for the codes in Table 1. We prefer $MAE = 1/181 \sum_{\theta=0^\circ}^{180^\circ} |\epsilon(\theta)|$ 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.

In overall it seems that the typical relative error for I for DDA codes is somewhere between 2% and 6% for silicate and ice type refractive indices and for particles with size $x_{\mathrm{eq}} \approx 5$. For P, the typical pure 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 *zdd* and almost as good with *ddscat*, and not so good with *sirri* and *adda*.

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. (1), the side length of the vessel that envelopes 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 for the CPU times and memory consumptions of the DDA codes in our study are presented in Table 2. 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 speed. The computer is intended for efficient parallel computing and if serial programs that use only one processor are executed, the performance is comparable to a modern desktop PC.

Table 1: Mean absolute errors (MAE) for the intensity (I) and the linear polarization ratio (P) for all the DDA codes with five geometries and two refractive indices. Errors for intensity are relative to the exact solution while errors for polarization are pure 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.

		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**	**0.9330**	1.130	*4.590*	*4.823*
	P	1.314	1.109	**0.727**	**0.527**	0.740	0.934	*3.263*	*1.818*
cylinder	I	5.373	4.743	2.627	2.926	**1.392**	**2.479**	*8.090*	*6.436*
	P	1.210	*2.216*	0.999	1.204	**0.570**	**0.805**	*2.370*	1.897
4-sphere	I	*3.203*	1.473	**1.099**	0.964	1.610	**0.791**	1.995	*1.772*
aggregate	P	1.830	1.882	**1.676**	**1.761**	1.967	*2.106*	*2.079*	1.780
50-sphere	I	6.473	2.987	**5.573**	2.925	6.193	**2.903**	*7.811*	*6.017*
aggregate	P	*2.026*	0.711	1.866	*0.834*	**1.480**	0.735	1.826	**0.690**

Table 2: CPU times and memory consumptions for the DDA codes for two example cases and with m_i refractive index.

	Cubic grid of 28^3 dipoles,1 orientation		Spheroid, 256 orientations	
	CPU-time per orientation (sec)	Memory per processor (Mbyte)	CPU-time per orientation (sec)	Memory per processor (Mbyte)
sirri	19	54.1	41	74.8
ddscat	36	22.5	30	32.3
zdd	85	21.6	51	**
adda	10	15.3	3	21.6

In the IBMSC environment, the latest version of *adda* seems to be by far the fastest. The *sirri* code is also quite fast in calculating single orientation. When averaging over several orientations, *sirri* will not perform that well. This is due to the fact that *sirri* uses random orientation angles distributed evenly on the unit sphere. The other codes use systematic sampling scheme for the orientation angles. The DDA computations can be divided into two stages. First one is when the field induced on each dipole is calculated. Second one corresponds to the determination of the scattered field in far zone. Since observational conditions do not influence the first stage that consumes a lot of computation time, codes with systematic sampling approach can use the first stage results to calculate scattering properties in a few different scattering planes with a little extra CPU time costs. *Sirri* and *adda* use double precision floats which shows in the large memory consumption of *sirri*. However, larger accuracy in number presentation does not seem to help *sirri* with the accuracy of the results. 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).

All the codes compared here have some good qualities and also some drawbacks. For example, *zdd* and *ddscat* 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* is freely available. On the other hand, *ddscat* can not use dynamical memory allocation and user-specified orientation angles, *sirri* has poor accuracy with polarization in the backscattering domain, etc.

A more detailed article of this study is under preparation and will hopefully be published soon.

Acknowledgments

A. Penttilä is thankful for the financial support of the Finnish Cultural Foundation and the Magnus Ehrnrooth foundation. 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] H. C. van de Hulst, *Multiple light scattering*, (Academic Press, New York 1980).

[2] M. I. Mishchenko and L. D. Travis, "Capabilities and limitations of a current FORTRAN implementation of the T-matrix method for randomly oriented, rotationally symmetric scatterers", J. Quant. Spectrosc. Radiat. Transfer, **60**, 309–324 (1998).

[3] D. W. Mackowski and M. I. Mishchenko, "Calculation of the T matrix and the scattering matrix for ensembles of spheres", J. Opt. Soc. Am. **A13**, 2266–2278 (1996).

[4] B. T. Draine and P. J. Flatau, User Guide for the Discrete Dipole Approximation Code DDSCAT (Version 6.1) `http://arxiv.org/abs/astro-ph/0409262` (2004).