A comparative study of efficient iterative solvers for the discrete dipole approximation

Patrick Chaumet

A comparative study of efficient iterative solvers for the discrete dipole approximation

Patrick C. Chaumet

[a]*Institut Fresnel, Aix Marseille Univ, CNRS, Centrale Marseille, Avenue Escadrille Normandie Niemen, Marseille, 13013, France*

Abstract

The discrete dipole approximation (DDA) is used to compute the electromagnetic diffraction of a three-dimensional object. Computationally, the DDA involves solving large, dense systems of linear equations through iterative methods such as QMR, GPBiCG and BiCGstab. In this paper, we propose to study two new methods (IDR(s) and GPBiCGstab(L)) for objects larger than the wavelength of illumination. We show that while IDR(s) can present a reduced computation time compared to other methods, it may not converge in some cases. Conversely, GPBiCGstab(L) always converges and also has a reduced computation time compared to QMR, GPBiCG and BiCGstab.

Keywords: Discrete dipole approximation, Iterative method, IDR(s), GPBiCGstab(L)
2008 MSC: 35Q60, 65F10

1. Introduction

The discrete dipole approximation (DDA) is a method used to compute the electromagnetic diffraction for three-dimensional objects. For more information, please refer to Refs. [1, 2] or more recent references such as Refs. [3, 4]. While DDA is a simple and versatile tool, its main weakness is having to solve a linear system of size ($3N\times3N$), where N is the number of discretization elements of the objet under study. For objects larger than the wavelength of illumination, this can be more than a million, with in addition, a matrix with the property of being dense [5]. It is clear that inverting such a matrix is impossible. Therefore, Purcell *et al.* [1] suggested solving the linear equation system using a simplistic iterative method, while Draine proposed the use of a conjugate gradient method [2]. In the literature there are numerous iterative methods, but it is impossible to say which one is the best because it depends on the considered matrix. However, all of them require performing many matrix vector product (MVP) to converge on the correct solution. Goodman *et al.* have shown that, thanks to the block Toeplitz structure of the matrix, the MVPs could be performed quickly using three-dimensional fast Fourier transforms (FFT) [6], but it remains nevertheless essential to find the most suitable iterative method to obtain the result as quickly as possible. A bad iterative method for the DDA can result in a long calculation times, sometimes taking hours, or the method may not converge at all. Today, the iterative methods used for the DDA are often based on Krylov subspace [7], which are well adapted for solving linear systems with nonsymmetric matrices [8, 9, 10, 11]. Currently, the most used iterative method for the DDA are the quasi-minimal residual, the stabilized version of the biconjugate gradient and

the Generalized Product Bi-Conjugate Gradient. In this article, we compare to these three iterative methods, two other iterative methods: the induced dimension reduction (IDR) [12], which is a method which has never been tested before for the DDA, and the generalized product-type methods based on bi-conjugate gradient (GPBiCG) which uses a novel stabilizing polynomials of degree L, which is a very recently published method [13, 14].

2. The discrete dipole approximation

2.1. Principle of the DDA

As the DDA has been previously presented, we will only briefly review the method's principle. More details can be found in Ref. [4]. The object is discretized into a set of N small cubic subunits of size d. Under the influence of the incident wave, each subunit is polarized and radiates an electromagnetic field. To determine the field at each subunit position, taking into account the coupling between each element of the discretization, we must solve a system of linear equations of the following form:

$$\mathbf{E} = \mathbf{E}_{\text{ref}} + \mathbf{A}D_{\alpha}\mathbf{E} \quad (1)$$

$$(\mathbf{I} - \mathbf{A}\mathbf{D}_{\alpha})\,\mathbf{E} = \mathbf{E}_{\text{ref}}, \quad (2)$$

where $\mathbf{A}$ is a matrix of size $(3N \times 3N)$ containing all the Green's tensors, [15] $\mathbf{E}$ and $\mathbf{E}_{\text{ref}}$ are $3N$ vectors containing the local and reference fields (*i.e.* filed in the absence of the object) at the position of each element of discretization. $\mathbf{D}_{\alpha}$ is a diagonal matrix of size $(3N \times 3N)$ containing the polarizabilities of each subunits, and $\mathbf{I}$ is the identity matrix of size $(3N \times 3N)$. Once Eq. (2) is solved, it is easy to quickly calculate the diffracted field in all space [16].

The cornerstone of the DDA is to quickly solve Eq. (2), knowing that the value of N can be very large (it may be larger than one million for objects larger than the wavelength of illumination) and $\mathbf{A}$ being a dense matrix. Due to the size of the matrix, compute the inverse of $(\mathbf{I} - \mathbf{A}\mathbf{D}_{\alpha})$ is not possible. Therefore, the solution is to use an iterative method to solve the linear system $\overline{\mathbf{A}}\mathbf{E} = \mathbf{E}_{\text{ref}}$, where $\overline{\mathbf{A}} = (\mathbf{I} - \mathbf{A}D_{\alpha})$.

2.2. Solve iteratively the system of linear equations of the DDA

The principle of an iterative method for solving the system of linear equations $\overline{\mathbf{A}}\mathbf{E} = \mathbf{E}_{\text{ref}}$ is to create a sequence $\mathbf{E}_k$ such that:

$$r_k = \frac{||\overline{\mathbf{A}}\mathbf{E}_k - \mathbf{E}_{\text{ref}}||}{||\mathbf{E}_{\text{ref}}||}, \quad (3)$$

with the residue r_k tending towards zero when k increases, in which the k-th approximation is derived from the previous ones [17]. The iterative process is stopped when $r_k < \eta$, where the value of η is set by the user and depends on the desired precision of the field.

The iterative methods necessitate the execution of one or two matrix-vector products (MVP) during each iteration. To perform the MVP quickly, Draine *et al.* suggested using the 3D fast Fourier transform because the matrix is Toeplitz [18, 6]. Then, the total computation time to obtain the electromagnetic field will depend on two factors. Firstly, the computation time required to perform the MVP. Secondly, the computation time in the iterative method itself, which involves operations on vectors. This time will be multiplied by the number of iterations necessary to achieve the desired accuracy. It should be noted that the calculation of the MVP can also be

divided into two parts. The first part involves the calculation of FFT and inverse FFT, while the second part involves the product of these FFTs. Notice that the entire matrix $\overline{\mathbf{A}}$ is not stored in memory. Indeed, when a matrix is Toeplitz, we only need to store one row of the matrix. Using the symmetries of Green's tensor, we therefore need to store in memory only 6 vectors of size 8N (the eight comes from the fact that for the matrix-vector convolution product, we need to multiply the size of the Topelitz matrix by two in each direction of space).

The best iterative method for DDA is the method that, on the one hand, always converges and, on the other hand, obtains the result as quickly as possible, *i.e.* generally with the least possible number of MVPs. The best known iterative method is the conjugate gradient [19], which was the first one used for the DDA [2, 6]. In 1997, Flatau studied different iterative methods (conjugate gradient (CG), biconjugate gradient (BiCG), the stabilized version of BiCG (BiCGstab), the quasi-minimal residual (QMR), the transpose-free QMR. He concluded that the best iterative method was BiCGstab. In 1996, Rahola found that QMR was the best iterative method for the DDA, but the objects studied were small and weakly contrasted because they required less than 100 MVPs, whatever the iterative method chosen [20]. In 2006 Fan *et al.* compared QMR, GPBiCG, BiCGstab and BiCGstab(L) and found that QMR require fewer MVPs than GPBiCG and BiCGstab when $|\varepsilon| > 4$ [21]. More recently in 2007, Yurkin *et al.* studied three iterative methods (QMR, BiCG and BiCGstab) for particles much larger than the wavelength of illumination [3, 5] and conclude that QMR and BiCG were the best methods to use. In the case of magneto-dielectric particles, the most efficient method was the general product bi-conjugate gradient (GPBiCG) [22].

Finally, it appears that the methods generally used are QMR for the code done by Yurkin [5], GPBiCG for the idiot friendly discrete dipole approximation (IFDDA) code [22], and BiCGstab for the code done by Draine and Flatau [23]. These three codes are available for free.

In this article, we will compare the three iterative methods: QMR, GPBiCG and BiCGstab, with the Induced Dimension Reduction (IDR(s)) [12], a method that has never been tested for the DDA and GPBiCGstab(L) an iterative method developed very recently and introduced by Aihara [13, 14] which is based on a combination of GPBiCG and BiCGstab. These two algorithms have in common that they require the solution of a small system of linear equations within the iterative method. Notice that the algorithms of IDR(s) and GPBiCGstab(L) are given in Appendix A and Appendix B, respectively. The algorithms for QMR and BiCGstab are taken from the parallel iterative methods package [24] (be careful, because in the QMR and BiCGstab code, conjugated complexes are missing in some internal products). The algorithm for GPBiCG is from Ref. [25, 26]. We will particularly focus on objects with high permittivity and larger than the wavelength of illumination, which require a large number of MVPs, often exceeding 1000.

Note that all the calculations have been made with the software IFDDA [27] available for free at the following address: https://www.fresnel.fr/spip/spip.php?article2735&lang=fr. The routine IDR(s) and GPBiCGstab(L) are also available on the net in the IFDDA code. The reader can also find all the other iterative methods used in this article in the IFDDA code. All calculations were done with the processors: Intel(R) Xeon(R) Platinum 8168 CPU @ 2.70GHz, and all the code is written in FORTRAN parallelized with OpenMP on 24 processors and for the FFT we use the FFT in the West [28].

3. Results

3.1. Lossless spherical particle

We begin to study a lossless sphere of relative permittivity $\varepsilon = 3$ and radius a illuminated by a plane wave of wave number k_0. The chosen discretization is $d \approx \lambda/(6|n|)$, where $n = \sqrt{\varepsilon}$ is the refractive index of the object. Therefore, the number N of subunits changes according to $k_0 a$. The inset in the bottom right of the Fig. 1 indicates the values of N chosen for the sphere versus $k_0 a$. This gives $0.5 < k_0 d|n| < 1.05q$, enough discretization to have an accuracy of the order of 10% if we compute the extinction cross-section with the DDA. For example, for $k_0 a \in [20;\ 25]$ we have almost $N = 96^3 \approx 1$ million of dipoles, and for the largest sphere the error on the extinction cross section computed with the DDA compared to Mie theory is about 8%. It should also be noted that the number of iterations depends little on the discretization chosen, for more details, see Appendix C. We plot in Fig. 1(a), in log scale the number of MVPs versus the size parameter $k_0 a$. We can observe that the number of MVPs increases with $k_0 a$. This is due to the fact that with the increase of the size, the multiple scattering within the object becomes more significant, and consequently the spectrum of the matrix is broadened. To compare the efficiency of the two new methods introduced compared to the old ones, we decided to use GPBiCG as a reference and calculate the relative change (RC) in the number of MVP compared to GPBiCG:

$$\mathrm{RC}_{\mathrm{MVP}} = \frac{\text{Number of MVP}_{\text{method}} - \text{Number of MVP}_{\text{GPBiCG}}}{\text{Number of MVP}_{\text{GPBiCG}}}. \tag{4}$$

Obviously, the result is equal to zero for GPBiCG (black line). In Fig. 1(b), $\mathrm{RC}_{\mathrm{MVP}}$ is plotted versus the size parameter $k_0 a$. The BiCGstab is always close to GPBiCG, while QMR requires a higher number of MVPs. IDR(s) is clearly the best method as it significantly reduces the number of MVPs ($\approx 40\%$ for $s = 8$). The higher the value of s, the lower the number of MVPs required. GPBiCGstab(L) is between IDR(s) and GPBiCG. If we now consider the gain in computation time with GPBiCG as a reference, *i.e.*

$$\mathrm{RC}_t = \frac{\text{time}_{\text{method}} - \text{time}_{\text{GPBiCG}}}{\text{time}_{\text{GPBiCG}}}, \tag{5}$$

we can see that IDR(8) is no longer the best method, but it may even be one of the worst for large values of $k_0 a$. GPBiCGstab(L) is, for large values of $k_0 a$, the best method whatever the value of L. The value $L = 8$ gives the best result and is close to IDR(4). The reason for the substantial slowdown of IDR(8) is in its algorithm, and will be explained in the following paragraph. While for GPBiCGstab(L), the loss in computation time when L increases is less important. We tested BiCGstab(L) for L=2, 4, 8, (not plotted) but the result does not change with respect to the value of L and is always close to the GPBICG method.

Now, we study the evolution of the residue as a function of the number of MVPs for the iterative methods seen previously, for a sphere illuminated by a plane wave of size parameter $k_0 a = 20$ and relative permittivity of 3, see Fig. 2(a). Notice that with the chosen discretization, the error on the extinction cross section computed with the DDA compared to Mie theory is about 9%. QMR, GPBiCG and BiCGstab have similar behavior, with a slightly smoother curve for QMR. For GPBiCGstab(L), the gain, compared to the three historical methods, in terms of the number of MVPs is increasingly important as L increases. It is worth noting that after $r = 10^{-5}$, the slope becomes much steeper for GPBiCGstab(L). For IDR(s), the gain is even more spectacular, with more than a factor of 2 for IDR(8) with a very fast decrease of the curves. If we

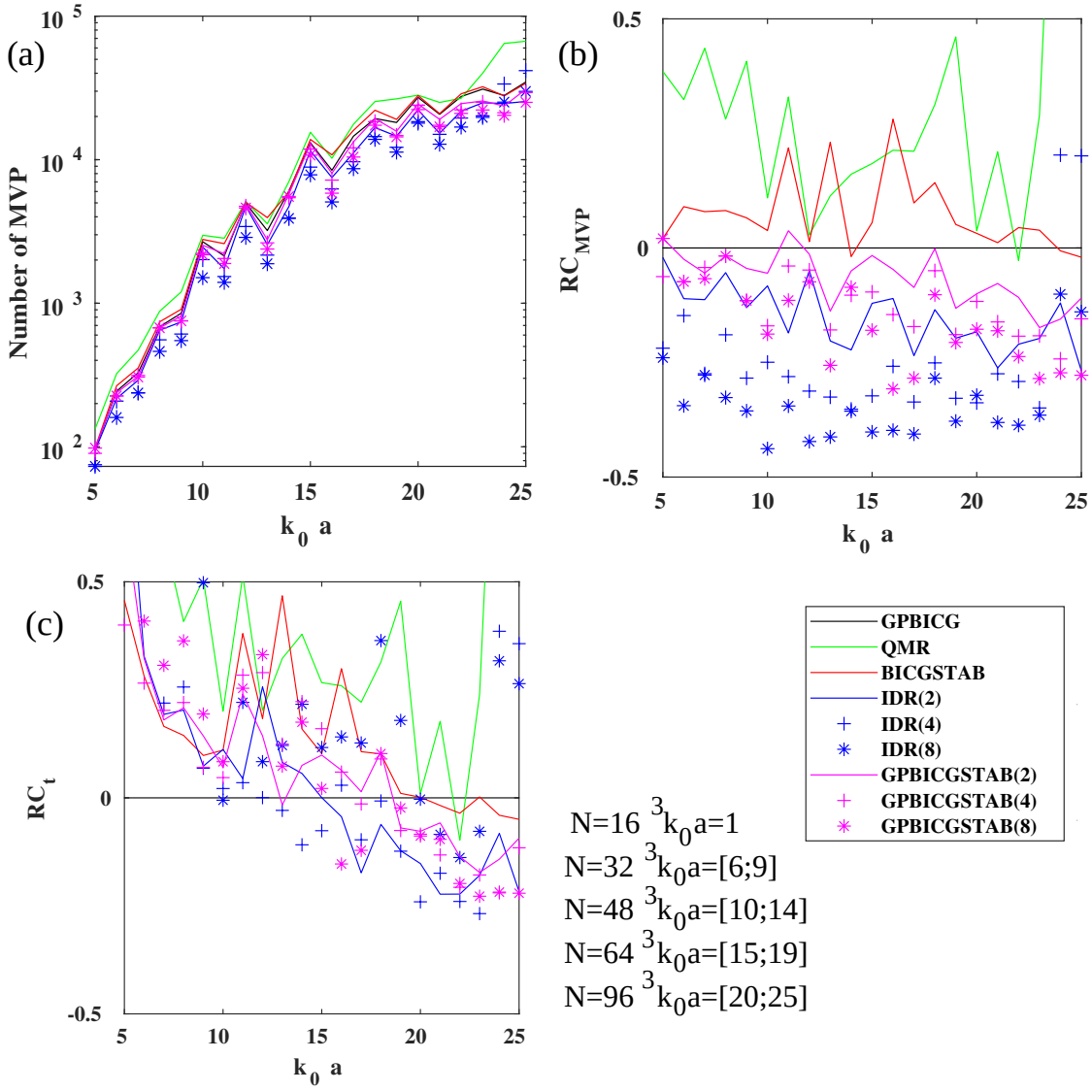

Figure 1: Sphere of relative permittivity $\varepsilon = 3$ illuminated by a plane wave. The stopping criteria of the iterative method is set to $\eta = 10^{-4}$. (a) Number of MVPs versus k_0a in log scale for the different iterative methods. (b) RC_{MVP} (relative change of the different iterative methods for the number of MVPs compared to GPBiCG) versus k_0a. RC_t (relative change of the different iterative methods for the computation time compared to GPBiCG) versus k_0a. The inset in the bottom right of the figure indicates the values of N chosen versus k_0a, N depending on sphere radius.

look at the same evolution of the residue but as a function of the computation time, Fig. 2(b), the gains are a slightly less significant for GPBiCGstab(L) and IDR(s) due to the internal computa-

tions within the iterative routine. For IDR(s), we can even see that IDR(8) becomes slower than IDR(4). Nevertheless, it is clear that GPBiCGstab(L) and IDR(s) are faster than the three usual iterative methods used in DDA codes. We have also tested BiCGstab(L) for L=2, 4, 8, but the result is always worse than the GPBICG method. To get a better understanding of the slowdown of GPBiCGstab(L) and IDR(s) with respect to s and L, respectively, in Tab. 1, we report the computation time in the iterative method and the computation time to compute the MVP (computation time of the FFTs plus the computation time of the product of the FFTs) for a sphere of relative permittivity $\varepsilon = 3$ with a size parameter $k_0 a = 20$ illuminated by a plane wave with $\eta = 10^{-4}$. It is clear that as s increases, the computation time spent in the IDR(s) routine becomes more and more consequent, which explains the slowing down of IDR(8). The same process is observed for GPBiCGstab(L), but to a lesser extent. This is because for IDR(s), the number of scalar products increases in s^2 for one MVP, while for GPBiCGstab(L), the number of scalar products for one MVP increases in L, see Appendix A and Appendix B for more details. For GPBiCG, QMR and BiCGstab, the computational time spent in the iterative routine is negligible compared to the computation time of the MVP.

Iterative method	Time in the iterative method	Time for the MVP	Total time	% in the iterative method
GPBiCG	437	7632	8069	5.4 %
QMR	298	7298	7596	3.9 %
BiCGstab	277	7536	7813	3.6 %
IDR(2)	676	5651	6327	10.7 %
IDR(4)	833	4041	4874	17.1 %
IDR(8)	1693	3458	5151	32.9 %
GPBiCGstab(2)	663	6352	7015	9.5 %
GPBiCGstab(4)	711	5793	6504	10.9 %
GPBiCGstab(8)	878	5098	5976	14.7 %

Table 1: Sphere of relative permittivity $\varepsilon = 3$ with a size parameter $k_0 a = 20$ illuminated by a plane wave with $\eta = 10^{-4}$. Time in second in the different section of the code.

In Tab. 2, we look at the efficiency of the parallelization in OpenMP of the iterative methods for a sphere with a size parameter $k_0 a = 10$. We observe that the gain in efficiency ranges from a factor of 5 to 6 when using 1 to 6 processors. With 24 processors, we have a gain of a factor of 17 compared to one processor, indicating highly efficient parallelization. The percentage of time spent in the iterative method slightly increases with the number of processors, meaning that the parallelization of FFTW is a bit more efficient than that of the iterative routines programmed in OpenMP, which is normal because FFTW uses MPI.

Note that we have tested other iterative methods, such as many variant of GPBiCG [29, 30, 26, 31] but they all give similar results to GPBiCG. In the same way, we have also tried transpose-free QMR [24], QMRBiCGstab [32], the conjugate orthogonal residual squared [33], and BiCGstab(L) [34] but they all converge much more slowly than QMR, BiCGstab, GPBiCG or do not converge at all. We also tested a Gaussian beam illumination with a waist of λ [35] centered at $(0, a/2, 0)$ and the conclusions that have been drawn in this subsection remain valid.

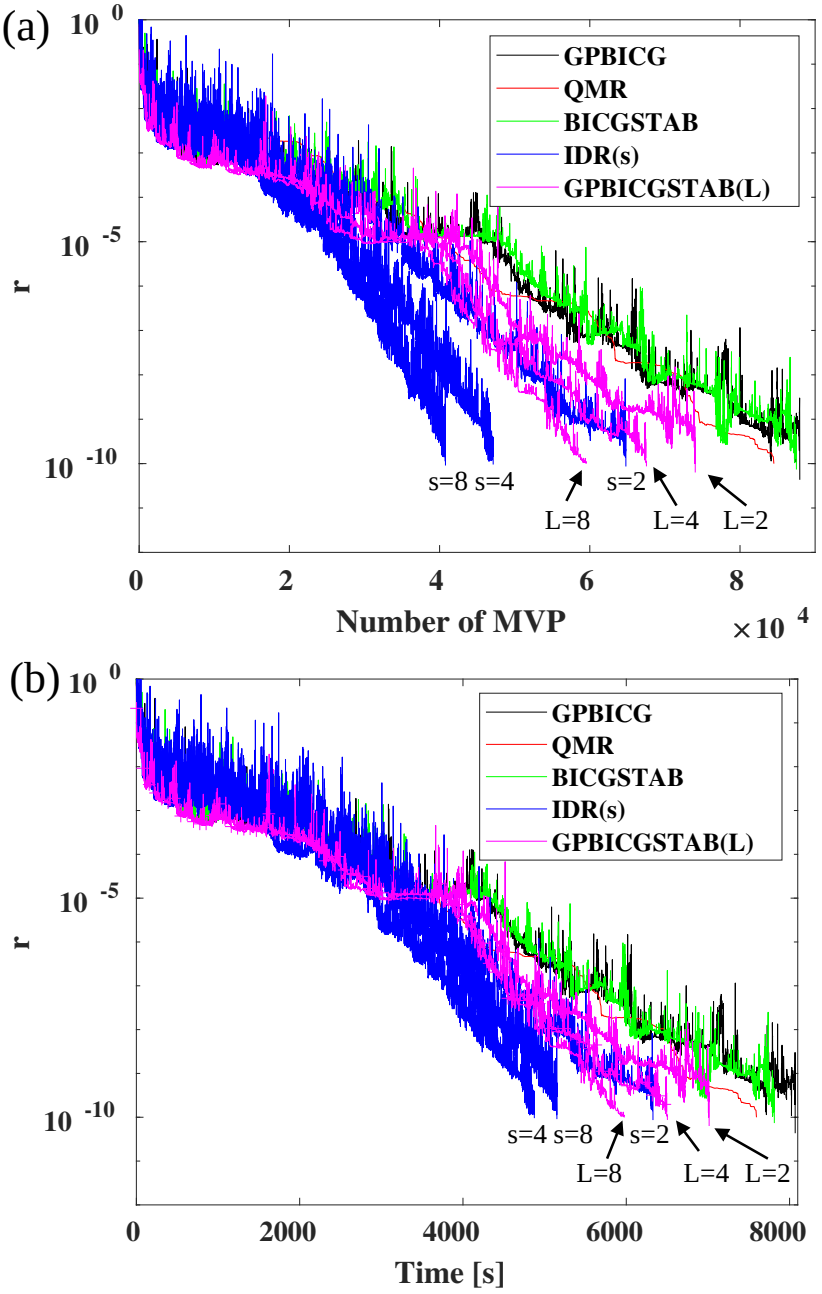

Figure 2: Sphere of relative permittivity $\varepsilon = 3$ with a size parameter $k_0 a = 20$ illuminated by a plane wave with $\eta = 10^{-10}$. (a) Residue versus the number of MVP. (b) Residue versus the time of computation.

Iterative method	Time in the iterative method			Time for the MVP			Total time			% in the iterative method		
Number of processor	24	6	1	24	6	1	24	6	1	24	6	1
GPBiCG	1.7	3.6	20.5	21.7	68.4	391	23.4	72.0	411	7.2	5.0	8.0
QMR	1.2	3.6	18.7	23.3	75.4	407	24.5	79.0	426	5.0	4.6	4.4
BiCGstab	0.9	2.2	12.3	22.1	72.1	367	23.0	74.3	379	4.1	3.0	3.2
IDR(2)	2.3	6.8	31.6	17.7	58.9	261	20.0	65.7	293	11.3	10.3	10.8
IDR(4)	4.0	11.3	61.4	15.9	49.4	263	19.9	60.7	324	20.2	18.6	18.9
IDR(8)	7.5	22.4	122.2	11.3	40.4	204	18.8	62.8	326	40.0	35.7	37.4
GPBiCGstab(2)	2.4	5.7	28.6	19.9	61.9	316	22.3	67.6	336	10.7	8.5	8.3
GPBiCGstab(4)	2.8	6.2	31.4	19.1	56.4	302	21.9	62.6	333	12.9	9.9	9.4
GPBiCGstab(8)	3.7	8.2	39.2	17.7	56.0	339	21.4	64.2	378	17.5	12.7	10.4

Table 2: Sphere of relative permittivity $\varepsilon = 3$ with a size parameter $k_0 a = 10$ illuminated by a plane wave with $\eta = 10^{-4}$. Time is second in the different section of the code depending on the number of processors used.

3.2. Silver particle

In this section, we study the behavior of iterative methods with a metallic sphere, see Fig. 3. We have chosen a silver sphere at $\lambda = 500$ nm with $\varepsilon = -8.5{+}0.76i$. The meshsize is always fixed to $d \approx \lambda/(6|n|)$. It should be noted that the extinction cross section computed with the DDA shows a relative error between 4 and 9% versus $k_0 a$, compared with the calculation made with Mie's theory. We have not represented QMR in Figs. 3(b) and 3(c) because $\mathrm{RC}_{\mathrm{MVP}}$ and RC_t are always around 1. We can notice that IDR(4) and IDR(8) do not converge when $k_0 a > 15$. GPBiCG and BiCGstab look very similar. GPBiCGstab(L) always converges and has a lower number of MVPs than GPBiCG and BiCGstab, as shown in Fig. 3(b). On the other hand, regarding the computation time, Fig. 3(c), for $k_0 a < 15$, GPBiCGstab(L) is much longer and for $k_0 a > 15$, we have $\mathrm{RC}_t \approx 0$. Unfortunately, the advantage in terms of MVP is not reflected in the computation time due to the time taken in the iterative routine.

3.3. Inhomogeneous Object

We consider an inhomogeneous cube of side $a = 12\lambda$ with discretization $N = 128{\times}128{\times}128$ ($d \approx \lambda/10$) with a random relative permittivity with variance σ^2 and defined as $\langle \varepsilon(\mathbf{r}), \varepsilon(\mathbf{r}') \rangle = 1 + \sigma^2 \exp\left(-\frac{\|\mathbf{r}-\mathbf{r}'\|^2}{l_c^2}\right)$ [36]. We have chosen $l_c = \lambda$ and in Tab. 3, we present the number of MVPs and the computation time in seconds for different values of σ. The higher the value of σ, the more inhomogeneous the object is and the more multiple scattering occurs. In this configuration, IDR(s) is not a suitable method. With the case $s = 8$, the residue will very quickly diverge to high values. BiCGstab does not converge on the highest σ, while QMR and GPBiCG converge for all values of σ, with GPBiCG being slightly faster. In this case, GPBiCGstab(L) is clearly the best method, with for the highest σ value a number of MVPs divided by 5 compared to GPBiCG for $L = 8$, while the time of computation is divided by 6.

3.4. Cuboid and preconditioner

Recently, Groth *et al.* introduced a multilevel circulant preconditioner [37, 38] to solve the linear system and improve the rate of convergence of the iterative method [39]. We recently showed that the multilevel circulant introduced by Groth was well suited for flat, homogeneous

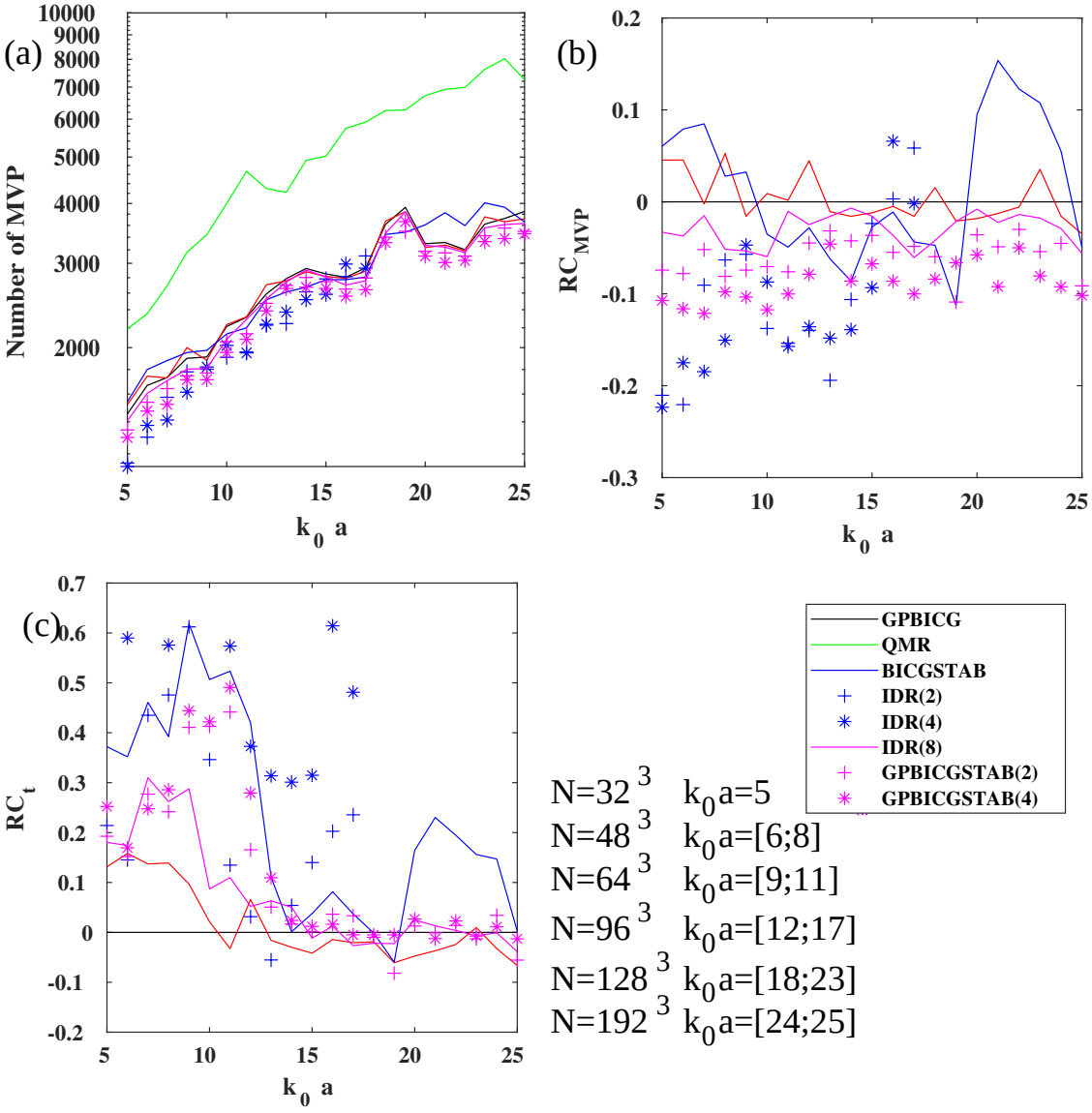

Figure 3: Sphere of silver illuminated by a plane wave with $\varepsilon = -8.5 + 0.76i$. The stopping criteria of the iterative method is set to $\eta = 10^{-4}$. (a) Number of MVP versus $k_0 a$ in log scale for the different iterative method. (b) $\mathrm{RC}_{\mathrm{MVP}}$ versus $k_0 a$. (c) RC_{t} versus $k_0 a$.

objects and for relative permittivities lower than 2 [40]. However, the preconditioned iterative method fails to converge when $\varepsilon > 2.5$ [40]. In this section, we study the number of MVPs for a flat, homogeneous cuboid of size $(20\lambda, 20\lambda, 2\lambda)$ with $d = \lambda/(7.5)$ without preconditioner and with the left and right preconditioner for different values of relative permittivities, as shown in

	$\sigma = 0.13$		$\sigma = 0.16$		$\sigma = 0.19$		$\sigma = 0.22$	
	MVP	t	MVP	t	MVP	t	MVP	t
GPBiCG	50	18	88	35	176	63.4	4930	1846
BiCGstab	54	19	102	39	202	75	-	-
QMR	95	39	159	59	275	93	10147	3667
IDR(2)	51	15	84	25	-	-	19332	4806
IDR(4)	68	23	138	41	483	134	-	-
IDR(8)	-	-	-	-	-	-	-	-
GPBiCGstab(2)	50	12	86	20	162	38	2334	568
GPBiCGstab(4)	50	12	82	20	138	34	1250	306
GPBiCGstab(8)	50	12	82	21	146	38	1106	284

Table 3: Number of MVPs and computation time (t) in second for the different iterative methods versus σ for $\eta = 10^{-4}$ and $l_c = \lambda$.

Tab. 4. As seen in Ref. [40], for $\varepsilon < 2.5$ using the preconditioner is very efficient in decreasing the number of MVPs by a factor of 10 and the computation time by a factor of 3. We see that the right preconditioner is a slightly better than the left preconditioner.

When $\varepsilon > 2.5$, the three methods QMR, BiCGstab and GPBiCG do not converge at all with the right or left preconditioner confirming what is said in Ref. [40]. On the other hand, IDR(s) always converges very quickly with the preconditioner and converges faster for larger s. This is also true for GPBiCGstab(L), but to a lesser extent because even for $L = 8$, the last value of ε does not converge with the preconditioner.

3.5. Spherical particle in presence of a multilayer system

The DDA can also compute electromagnetic wave diffraction for objects in the presence of multilayers. We chosen the configuration shown in Fig. 4, which consists of a sphere with a radius a placed between two glass planes separated by a distance $2a$. In this case, the matrix $\mathbf{A}$ becomes Toeplitz only in the x and y directions. Therefore, the product of the matrix $\mathbf{A}$ by the field $\mathbf{E}$ is done with 2D FFTs, thus slowing down the calculation of the MVP. Figure 5(a) shows the number of MVPs, Fig. 5(b) and 5(c) show $\mathrm{RC}_{\mathrm{MVP}}$ and RC_t, respectively, for a sphere of permittivity $\varepsilon = 3$ as a function of $k_0 a$ with $\eta = 10^{-4}$. QMR is, in this case, the method that requires the most MVP, while BiCGstab requires slightly more MVP than GPBiCG. GPBiCGSTAB(8) is a little better than GPBiCG, and clearly IDR(8) is the best method. We note that the curves in Fig. 5(b) are very similar to those in Fig. 5(c). The same comment can therefore be made about the computation time.

Figure 6 studies the evolution of the residue for a sphere with a size parameter of $k_0 a = 10$ and a relative permittivity of $\varepsilon = 3$. QMR for high residue joins the two methods GPBiCG and BiCGstab and confirms the fact that for a value of $\eta > 10^{-6}$ the three methods give similar results. GPBiCGSTAB(L) is better than the three previous methods, but the difference is small, while IDR(s) is clearly the best method with a significant and consistent improvement in terms of MVP and computation time. The explanation for why the curves RC_{MVP} and RC_t are so similar is that the computation time spent in the iterative method is always less than 1% of the total time, regardless of the method used. This is due to the fact that the MVP is significantly slowed down

		$\varepsilon = 2.1$		$\varepsilon = 2.4$		$\varepsilon = 2.7$		$\varepsilon = 3.0$	
method	**P**	MVP	t	MVP	t	MVP	t	MVP	t
GPBiCG	No	1690	62	4046	158	6764	254	18778	712
	LP	146	23	144	23	-	-	-	-
	LR	112	22	112	22	-	-	-	-
BiCGstab	No	1716	65	4380	164	6956	258	17446	650
	LP	154	29	460	70	-	-	-	-
	LR	116	22	170	28	-	-	-	-
IDR(2)	No	1770	71	4068	158	7186	280	11578	459
	LP	99	18	135	22	267	40	1980	277
	LR	105	20	120	30	213	47	871	132
IDR(4)	No	1678	72	3451	148	7738	331	12821	547
	LP	99	18	135	22	224	35	873	127
	LR	86	19	103	20	195	33	499	76
IDR(8)	No	1695	91	3956	211	6940	365	12306	651
	LP	100	19	129	24	202	43	660	116
	LR	66	16	103	22	183	34	571	91
GPBiCGstab(2)	No	1598	63	3982	156	6622	256	17614	688
	LP	134	29	142	24	-	-	-	-
	LR	114	21	106	20	-	-	-	-
GPBiCGstab(4)	No	1610	63	3802	151	6450	253	15170	595
	LP	146	23	122	20	2794	393	-	-
	LR	106	21	106	20	2386	336	-	-
GPBiCGstab(8)	No	1570	65	3666	149	6338	262	14290	594
	LP	114	19	130	21	1266	182	-	-
	LR	114	22	114	22	898	136	-	-

Table 4: The object under study is a cuboid of size $(20\lambda, 20\lambda, 2\lambda)$ illuminated by a plane wave. Number of MVPs and computation time (t) in second for the different iterative methods for $\eta = 10^{-4}$ with no preconditioner (NP), left preconditioner (LP) and right preconditioner (RP) versus ε.

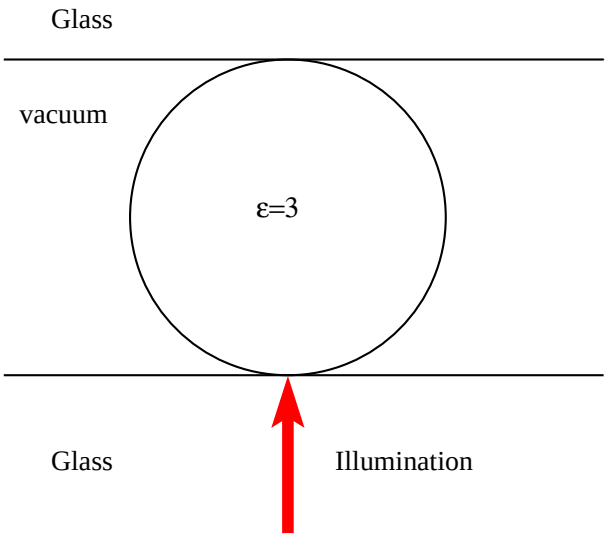

Figure 4: Sphere of relative permittivity $\varepsilon = 3$ with a radius a illuminated by a plane wave. The superstrate and the substrate are in glass ($\epsilon = 2.25$) separated by a distance $2a$.

by using only two dimensional FFTs. Therefore, the gain in MVP translates directly into a gain in time, contrary to objects in homogeneous space.

Now, we study the same configuration as the previous one, but with a silver sphere instead of a glass sphere for $\lambda = 500$ nm and $\eta = 10^{-4}$. Due to the similarity between RC_{MVP} and RC_t, only RC_t is plotted, as shown in Fig. 7. The conclusion is that IDR(s) is the best method with a gain of up to 40 % with $s = 8$, and GPBiCGSTAB(L) is also a good method with a gain of 20 % with $L = 8$ compared to GPBiCG. We also tried the same configuration with a superstrate in gold, and the results obtained were similar.

4. Conclusion

In this article, we have studied various iterative methods for solving the linear system of the DDA. We compared GPBiCG, BiCGstab and QMR, which are the most used methods for the DDA, with IDR(s) and GPBiCGstab(L). We observed that GPBiCG, BiCGstab and QMR exhibit similar convergence, with only QMR converging slightly slower when the stopping criterion value of the iterative method is fixed at $\eta = 10^{-4}$.

IDR(s) can be a very fast method, especially when preconditioning is used, but it may also not converge in some configurations. It is therefore complicated to advise to use IDR(s) as its behavior is difficult to predict.

Finally, we have seen that GPBiCGstab(L) always converges and requires less MVPs than GPBiCG, BiCGstab and QMR. In all cases where the object was much larger than the wavelength ($k_0 a > 15$), this method allowed us to save computation time. However, it should be noted that that this method requires the storage of many intermediate vectors and requires a bit more computation in the iterative method itself.

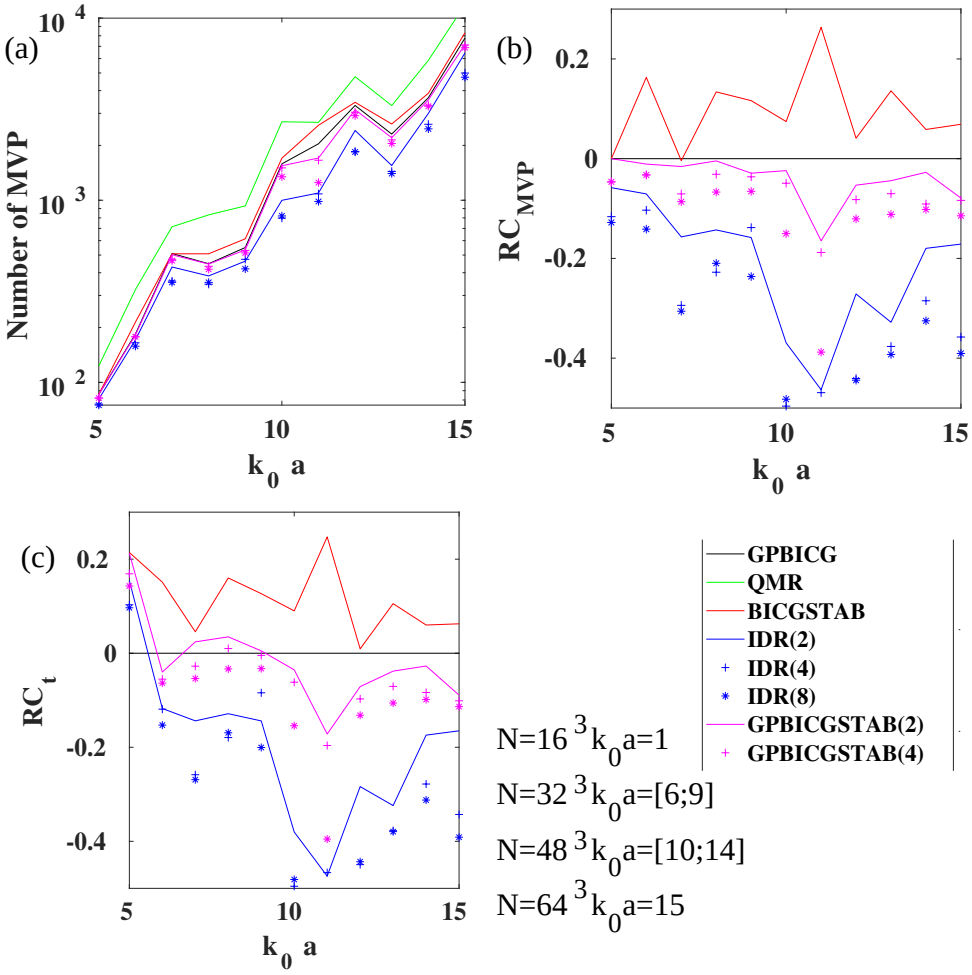

Figure 5: Geometrical configuration is described in Fig. 4. The stopping criteria of the iterative method is set to $\eta = 10^{-4}$. (a) Number of MVP versus k_0a in log scale for the different iterative method. (b) RC_{MVP} versus k_0a. (c) RC_t versus k_0a.

Appendix A. The IDR(s) algorithm

The IDR(s) algorithm has been introduced in Ref. [12] and a more comprehensive algorithm is given in Ref. [42]. For the convenient of the reader, the IDR(s) algorithm is briefly presented below:

1. Select an initial guess $\mathbf{x}$
2. Compute $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}$ Compute MVP
3. for $n = 0, \cdots, s-1$
4. $\mathbf{v}_n = \mathbf{A}\mathbf{r}_n$

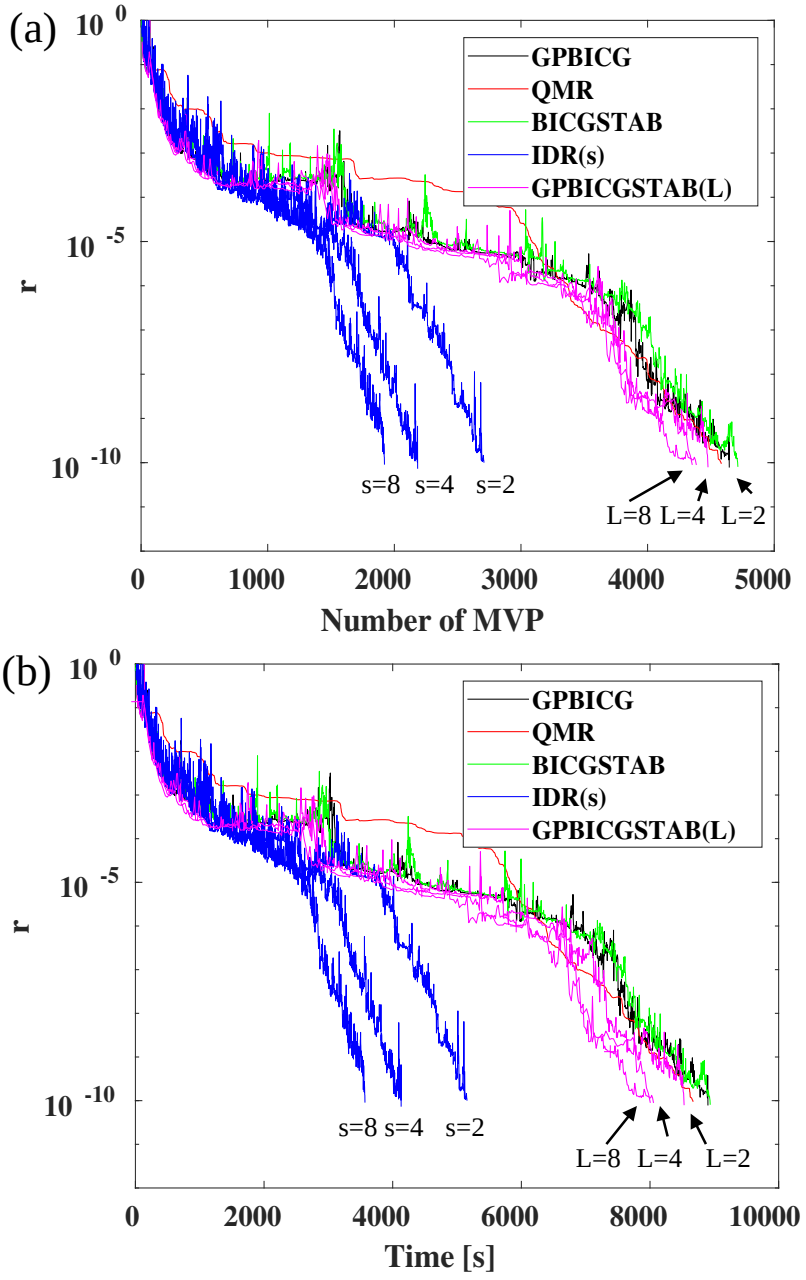

Figure 6: Sphere of relative permittivity $\varepsilon = 3$ with a size parameter $k_0 a = 10$ illuminated by a plane wave with $\eta = 10^{-10}$. (a) Residue versus the number of MVP. (b) Residue versus the time of computation.

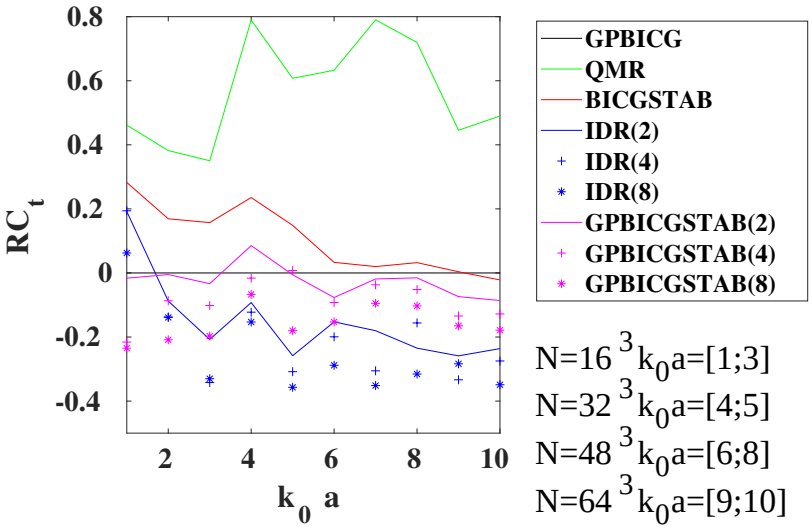

Figure 7: Sphere of silver with $\eta = 10^{-4}$ in the configuration described in Fig. 5. RC_t versus $k_0 a$.

5. $\omega = (\mathbf{v}_n, \mathbf{r}_n)/(\mathbf{v}_n, \mathbf{v}_n)$
6. $\mathbf{q}_n = \omega \mathbf{r}_n$; $\mathbf{e}_n = -\omega \mathbf{v}_n$
7. $\mathbf{r}_{n+1} = \mathbf{r}_n + \mathbf{e}_n$; $\mathbf{x}_{n+1} = \mathbf{x}_n + \mathbf{q}_n$
8. end;
9. $\mathbf{E}_s = (\mathbf{e}_{s-1}, \cdots, \mathbf{e}_0)$; $\mathbf{Q}_s = (\mathbf{q}_{s-1}, \cdots, \mathbf{q}_0)$
10. $\mathbf{P} = (\mathbf{p}_1, \cdots, \mathbf{p}_s)$ every entries of $\mathbf{p}_i$ are random complex numbers between 0 and 1.
11. $\mathbf{P}$ is orthonormalized with Gram-Schmidt method as $(\mathbf{p}_i, \mathbf{p}_j) = \delta_{ij}$
12. $n = s$
13. while $\|\mathbf{r}_n\|/\|\mathbf{b}\| >$ tol
14. Solve $\mathbf{c}_n$ from $\mathbf{P}^T \mathbf{E}_n \mathbf{c}_n = \mathbf{P}^T \mathbf{r}_n$
15. $\mathbf{v}_n = \mathbf{r}_n - \mathbf{E}_n \mathbf{c}_n$
16. If mod$(n, s+1) = s$ then
17. $\mathbf{t}_n = \mathbf{A}\mathbf{r}_n$ Compute MVP
18. $\omega_n = \frac{(\mathbf{t}_n, \mathbf{v}_n)}{(\mathbf{v}_n, \mathbf{v}_n)}$
19. $\mathbf{e}_n = -\mathbf{E}_n \mathbf{c}_n - \omega_n \mathbf{t}_n$
20. $\mathbf{q}_n = -\mathbf{Q}_n \mathbf{c}_n + \omega_n \mathbf{v}_n$
21. Else
22. $\mathbf{q}_n = -\mathbf{Q}_n \mathbf{c}_n$
23. $\mathbf{e}_n = -\mathbf{A}\mathbf{q}_n$ Compute MVP
24. End If
25. $\mathbf{r}_{n+1} = \mathbf{r}_n + \mathbf{e}_n$
26. $\mathbf{x}_{n+1} = \mathbf{x}_n + \mathbf{q}_n$
27. $\mathbf{E}_n = (\mathbf{e}_{n-1}, \cdots, \mathbf{e}_{n-s})$

28. $\mathbf{Q}_n = (\mathbf{q}_{n-1}, \cdots, \mathbf{q}_{n-s})$
29. $n = n + 1$
30. End While

$\mathbf{P}^T$ is the conjugate transpose matrix and $(\mathbf{a}, \mathbf{b}) = \mathbf{a}^T\mathbf{b}$ is the scalar product. Note that s represents the number of previous search directions that are retained and used to construct the new search direction in each iteration of the algorithm. Hence, in the IDR(s) algorithm we have to solve a system of linear equations of size $s \times s$ which is not parallelized because the size of the matrix is small. All other calculations are parallelized using OpenMP. Note that for one iteration of the algorithm, we have to perform 1 MVP and $s^2 + 4s + 2$ scalar products. The number of vectors needed for the algorithm is $5 + 3s$.

Note that in Ref. [42], a variant is proposed to calculate the ω factor. Our tests have shown that in our configuration, this modification always deteriorated the convergence of the algorithm.

Appendix B. The GPBiCGstab(L) algorithm

The GPBiCGstab(L) algorithm has been developed by Aihara [13, 14]. He proposes a stabilizing polynomial of degree L that combines the stabilizing polynomial of BiCGstab(L) and the polynomial given by a three term recurrence of GPBiCG. The algorithm given by Aihara is in a compact notation. For ease of understanding for the reader, we have rewritten it in this appendix in a form that is directly programmable in FORTRAN or C++. We want to solve $\mathbf{Ax} = \mathbf{b}$ with an initial guess and a tolerance fixed at tol:

1. Select an initial guess $\mathbf{x}$.
2. Compute the residue $\mathbf{r}_0 = \mathbf{b} - \mathbf{Ax}$ and choose a vector $\tilde{\mathbf{r}}_0$.
3. $\mathbf{p}_0 = \mathbf{r}_0$
4. $\rho = (\tilde{\mathbf{r}}_0, \mathbf{r}_0)$
5. for $j = 1, \cdots, L$
6. $\quad \mathbf{p}_j = \mathbf{A}\mathbf{p}_{j-1}$ Compute MVP
7. $\quad \sigma = (\tilde{\mathbf{r}}_0, \mathbf{p}_j)$
8. $\quad \alpha = \rho/\sigma$
9. $\quad \mathbf{x} = \mathbf{x} + \alpha\mathbf{p}_0$
10. $\quad$ for $i = 0, \cdots, j-1$; $\mathbf{r}_i = \mathbf{r}_i - \alpha\mathbf{p}_{i+1}$; end
11. $\quad \mathbf{r}_j = \mathbf{A}\mathbf{r}_{j-1}$ Compute MVP
12. $\quad \rho = (\tilde{\mathbf{r}}_0, \mathbf{r}_j)$
13. $\quad \beta = \rho/\sigma$
14. $\quad$ for $i = 0, \cdots, j$; $\mathbf{p}_i = \mathbf{r}_i - \beta\mathbf{p}_i$; end
15. end
16. $\mathbf{r}' = \mathbf{r}_0$; $\mathbf{p}' = \mathbf{p}_0$
17. for $i = 0, \cdots, L-2$; $\mathbf{s}_i = \mathbf{r}_{i+1}$; end
18. for $i = 0, \cdots, L-1$; $\mathbf{q}_i = \mathbf{p}_{i+1}$; end

Compute $\boldsymbol{\zeta}$ to minimize $\|\mathbf{r}_0 - [\mathbf{r}_1, \cdots, \mathbf{r}_L]\boldsymbol{\zeta}\|$ with $\boldsymbol{\zeta} = [\zeta_1, \cdots, \zeta_L]$:

19. $\mathbf{M} = [\mathbf{r}_1, \cdots, \mathbf{r}_L]$; Solve $\mathbf{M}^T\mathbf{M}\boldsymbol{\zeta} = \mathbf{M}^T\mathbf{r}_0$; size $L \times L$
20. $\mathbf{z} = \mathbf{0}$; for $i = 1, \cdots, L$; $\mathbf{z} = \mathbf{z} + \zeta_i\mathbf{r}_{i-1}$; end

21. $\mathbf{x} = \mathbf{x} + \mathbf{z}$ Update the solution
22. for $i = 1, \cdots, L$; $\mathbf{r}_0 = \mathbf{r}_0 - \zeta_i \mathbf{r}_i$; end
23. for $i = 1, \cdots, L$; $\mathbf{p}_0 = \mathbf{p}_0 - \zeta_i \mathbf{p}_i$; end

End of the initialization, the process of iteration begins

24. while $\|\mathbf{r}_0\|/\|\mathbf{b}\| > \text{tol}$
25. $\mathbf{y} = \mathbf{r}' - \mathbf{r}_0$; $\mathbf{u} = \mathbf{p}' - \mathbf{p}_0$
26. $\rho = (\tilde{\mathbf{r}}_0, \mathbf{r}_0)$
27. for $j = 1, \cdots, L$
28. if $j > 1$ then
29. for $i = 0, \cdots, L - j$; $\mathbf{s}_i = \mathbf{s}_i - \alpha \mathbf{q}_{i+1}$; end
30. for $i = 0, \cdots, L - j$; $\mathbf{q}_i = \mathbf{s}_i - \beta \mathbf{q}_i$; end
31. end
32. $\mathbf{p}_j = \mathbf{A}\mathbf{p}_{j-1}$ Compute MVP
33. $\mathbf{v} = \mathbf{q}_0 - \mathbf{p}_1$
34. $\sigma = (\tilde{\mathbf{r}}_0, \mathbf{p}_j)$, $\alpha = \rho/\sigma$.
35. $\mathbf{x} = \mathbf{x} + \alpha \mathbf{p}_0$, $\mathbf{z} = \mathbf{z} - \alpha \mathbf{u}$, $\mathbf{y} = \mathbf{y} - \alpha \mathbf{v}$
36. for $i = 0, \cdots, j - 1$; $\mathbf{r}_i = \mathbf{r}_i - \alpha \mathbf{p}_{i+1}$; end
37. $\mathbf{r}_j = \mathbf{A}\mathbf{r}_{j-1}$ Compute MVP
38. $\rho = (\tilde{\mathbf{r}}_0, \mathbf{r}_j), \beta = \rho/\sigma$
39. for $i = 0, \cdots, j$; $\mathbf{p}_i = \mathbf{r}_i - \beta \mathbf{p}_i$; end
40. $\mathbf{u} = \mathbf{y} - \beta \mathbf{u}$
41. end
42. Set $\mathbf{r}' = \mathbf{r}_0$, $\mathbf{p}' = \mathbf{p}_0$
43. for $i = 0, \cdots, L - 2$; $\mathbf{s}_i = \mathbf{r}_{i+1}$; end
44. for $i = 0, \cdots, L - 1$; $\mathbf{q}_i = \mathbf{p}_{i+1}$; end

Compute $\boldsymbol{\zeta}$ and $\mathbf{y}$ to minimize $\|\mathbf{r}_0 - [\mathbf{r}_1, \cdots, \mathbf{r}_L]\boldsymbol{\zeta} - \eta \mathbf{y}\|$:

45. $\mathbf{M} = [\mathbf{r}_1, \cdots, \mathbf{r}_L, \mathbf{y}]$; Solve $\mathbf{M}^T\mathbf{M} \begin{pmatrix} \boldsymbol{\zeta} \\ \eta \end{pmatrix} = \mathbf{M}^T \mathbf{r}_0$; size $(L + 1) \times (L + 1)$
46. $\mathbf{z} = \eta \mathbf{z}$; for $i = 1, \cdots, L$; $\mathbf{z} = \mathbf{z} + \zeta_i \mathbf{r}_{i-1}$; end
47. $\mathbf{x} = \mathbf{x} + \mathbf{z}$ Update the solution
48. Set $\mathbf{r}_0 = \mathbf{r}_0 - \eta \mathbf{y}$
49. for $i = 1, \cdots, L$; $\mathbf{r}_0 = \mathbf{r}_0 - \zeta_i \mathbf{r}_i$; end
50. Set $\mathbf{p}_0 = \mathbf{p}_0 - \eta \mathbf{u}$
51. for $i = 1, \cdots, L$; $\mathbf{p}_0 = \mathbf{p}_0 - \zeta_i \mathbf{p}_i$; end
52. end while

The number of vectors needed for the algorithm is $4L + 8$. In this configuration, all calculations are parallelized using OpenMP, except the matrix inversion. Note that the calculation of $\mathbf{M}^T\mathbf{M}$ requires only $(L + 2)(L + 1)/2$ scalar products because $\mathbf{M}^T\mathbf{M}$ is a Hermitian matrix. For one iteration of the algorithm, we need L MVP and $3L + 2 + (L + 2)(L + 1)/2$ scalar products. Hence, for 1 MVP, we have approximately $3 + L/2$ scalar products.

If we compare IDR(s) and GPBiCGstab(L), then we notice that for one MVP, the number of scalar products increase in s^2 for IDR(s) and in L for GPBiCGstab(L). This explains the significant growth of the time spent in the iterative method as a function of s compared to L, as shown in Tab. 1.

Appendix C. Influence of the discretization on the number of iteration

The number of subunits representing the object has little influence on the number of iterations required for the iterative method to converge [20, 41]. To check that this is true whatever the iterative method chosen, we choose the configuration of Fig. 1 with $k_0a = 6$ by varying the number N of dipoles representing the sphere. Table C.5 presents the evolution of the number of iterations; the relative error between the extinction cross-section computed with DDA and that calculated with Mie's theory; the factor $k_0|n|d$, as a function of N. We confirm that the three

Discretization	$N = 32^3$	$N = 48^3$	$N = 64^3$	$N = 96^3$	$N = 128^3$
GPBiCG	240	238	236	238	244
QMR	321	305	307	297	275
BiCGstab	256	228	234	240	232
IDR(2)	219	234	202	207	208
IDR(4)	208	183	181	163	348
IDR(8)	162	175	174	147	-
GPBiCGstab(2)	226	226	226	230	230
GPBiCGstab(4)	226	226	226	226	226
GPBiCGstab(8)	226	226	226	210	226
C_{ext} (%)	6.1	4.7	2.4	1.0	0.1
$k_0\|n\|d$	0.65	0.44	0.33	0.22	0.16

Table C.5: Evolution of the number of iteration for the different iterative method versus N for a sphere of permittivity $\varepsilon = 3$ and $k_0a = 6$. C_{ext} (%) gives the error between the extinction cross section computed with the DDA and the Mie's theory. The last line presents $k_0|n|d$ versus N.

methods usually chosen (GPBiCG, QMR, BiCGstab) have a number of iterations that varies little with the discretization. IDR(s) has also a stable number of iterations, but can sometimes exhibit instability. On the other hand, GPBiCGstab(L) is highly stable as a function of N. Note that if we change the radius of the sphere to choose a Mie resonance, then the number of iteration can depend strongly on N [20].

References

[1] E. M. Purcell, C. R. Pennypacker, Scattering and absorption of light by nonspherical dielectric grains, Astrophys. J. 186 (1973) 705–714.

[2] B. T. Draine, The discrete-dipole approximation and its application to interstellar graphite grains, Astrophys. J. 333 (1988) 848–872.

[3] M. A. Yurkin, A. G. Hoekstra, The discrete dipole approximation: An overview and recent developments, J. Quant. Spect. Rad. Transf. 106 (2007) 558–589.

[4] P. C. Chaumet, The discrete dipole approximation: A review, Mathematics 10 (2022).

[5] M. A. Yurkin, V. P. Maltsev, A. G. Hoekstra, The discrete dipole approximation for simulation of light scattering by particles much larger than the wavelength, J. Quant. Spect. Rad. Transf. 106 (2007) 546–557.

[6] J. J. Goodman, P. J. Flatau, Application of fast-fourier-transform techniques to the discrete-dipole approximation, Opt. Lett. 16 (2002) 1198–1200.

[7] T. Sogabe, Krylov Subspace Methods for Linear Systems, Springer Singapore, 2023.

[8] G. Meurant, J. D. Tebbens, Krylov Methods for Nonsymmetric Linear Systems, Springer Cham, 2020.

[9] K. Abe, G. L. Sleijpen, Bicr variants of the hybrid bicg methods for solving linear systems with nonsymmetric matrices, Journal of Computational and Applied Mathematics 234 (2010) 985–994. proceedings of the Thirteenth International Congress on Computational and Applied Mathematics (ICCAM-2008), Ghent, Belgium, 7–11 July, 2008.

[10] T. Sogabe, M. Sugihara, S.-L. Zhang, An extension of the conjugate residual method to nonsymmetric linear systems, Journal of Computational and Applied Mathematics 226 (2009) 103–113. special Issue: The First International Conference on Numerical Algebra and Scientific Computing (NASC06).
[11] K. Aihara, Variants of the groupwise update strategy for short-recurrence krylov subspace methods, Numerical Algorithms 75 (2017) 397–412.
[12] P. Wesseling, P. Sonneveld, Numerical experiments with a multiple grid and a preconditioned Lanczos type method, Springer Berlin Heidelberg, Berlin, Heidelberg, 1980.
[13] K. Aihara, Gpbi-cgstab(l): A lanczos-type product method unifying bi-cgstab(l) and gpbi-cg, Numerical Linear Algebra with Applications 27 (2020) e2298.
[14] I. Horiuchi, K. Aihara, T. Suzuki, E. Ishiwata, Global gpbicgstab(l) method for solving linear matrix equations, Numerical Algorithms (2022).
[15] J. D. Jackson, Classical Electrodynamics, 2nd ed. ed., Wiley, 1975.
[16] P. C. Chaumet, T. Zhang, A. Sentenac, Fast far-field calculation in the discrete dipole approximation, Journal of Quantitative Spectroscopy and Radiative Transfer 165 (2015) 88 – 92.
[17] A. Greenbaum, Iterative Methods for Solving Linear Systems, Society for Industrial and Applied Mathematics, 1997.
[18] P. J. Flatau, G. L. Stephens, B. T. Draine, Light scattering by rectangular solids in the discrete-dipole approximation: a new algorithm exploiting the Block-Toeplitz structure, J. Opt. Soc. Am. A 7 (1990) 593–600.
[19] M. R. Hestenes, E. Stiefel, Methods of conjugate gradients for solving linear system, J. of Research of the National Bureau of Standars 49 (1952) 409–436.
[20] J. Rahola, Solution of dense systems of linear equations in the discrete-dipole approximation, SIAM Journal on Scientific Computing 17 (1996) 78–89.
[21] Z. H. Fan, D. X. Wang, R. S. Chan, E. K. N. Yung, The application of iterative solvers in discrete dipole approximation method for computing electromagnetic scattering, Microwave Opt. Technol. Lett. 48 (2006) 1741–1746.
[22] P. C. Chaumet, A. Rahmani, Efficient iterative solution of the discrete dipole approximation for magneto-dielectric scatterers , Opt. Lett. 34 (2009) 917–919.
[23] P. J. Flatau, Improvements in the discrete-dipole approximation method of computing scattering and absorption, Opt. Lett. 22 (1997) 1205–1207.
[24] R. D. Da Cunha, T. Hopkins, The Parallel Iterative Methods (PIM) package for the solution of systems of linear equations on parallel computers, Appl. Numer. Math. 19 (1995) 33–50.
[25] S.-L. Zhang, Gpbi-cg: Generalized product-type methods based on bi-cg for solving nonsymmetric linear systems, SIAM Journal on Scientific Computing 18 (1997) 537–551.
[26] M. Thuthu, S. Fujino, Y. Onoue, An advanced iterative method based on intelligent determination of recurrences, IMECS 1 (2009).
[27] P. C. Chaumet, D. Sentenac, G. Maire, M. Rasedujjaman, T. Zhang, A. Sentenac, Ifdda, an easy-to-use code for simulating the field scattered by 3d inhomogeneous objects in a stratified medium: tutorial, J. Opt. Soc. Am. A 38 (2021) 1841–1852.
[28] M. Frigo, S. G. Johnson, The design and implementation of FFTW3, Proceedings of the IEEE 93 (2005) 216–231. Special issue on "Program Generation, Optimization, and Plat form Adaptation".
[29] S. Fujino, T. Sekimoto, Performance evaluation of gpbicgsafe method without reverse-ordered recurrence for realistic problems, IMECS 2 (2012).
[30] S. Fujino, A proposal of gpbicg-plus method, 15th MASCOT, 19th IMACS World Congress (2013).
[31] L. Zhao, T.-Z. Huang, Y.-F. Jing, L.-J. Deng, A generalized product-type bicor method and its application in signal deconvolution, Computers & Mathematics with Applications 66 (2013) 1372 – 1388.
[32] T. F. Chan, E. Gallopoulos, V. Simoncini, T. Szeto, C. H. Tong, A Quasi-Minimal Residual Variant of the Bi-CGSTAB Algorithm for Nonsymmetric Systems, SIAM J. Sci. Comput. 15 (1994) 338–347.
[33] B. Carpentieri, Y.-F. Jing, T.-Z. Huang, W.-C. Pi, X.-Q. Sheng, A novel family of iterative solvers for method of moments discretizations of maxwell's equations, Computational Electromagnetics International Workshop (2011) 85–90.
[34] G. L. G. Sleijpen, D. R. Fokkema, Bicgstab(l) for linear equations involving unsymmetric matrices with complex spectrum, Electronic Transactions on Numerical Analysis 1 (1993) 11–32.
[35] P. C. Chaumet, Fully vectorial highly non paraxial beam close to the waist., J. Opt. Soc. Am. A 23 (2006) 3197–3202.
[36] T. Zhang, P. C. Chaumet, E. Mudry, A. Sentenac, K. Belkebir, Electromagnetic wave imaging of targets buried in a cluttered medium u sing a hybrid inversion-dort method, Inverse Probl. 28 (2012) 125008.
[37] R. H. Chan, J. G. Nagy, R. J. Plemmons, Fft-based preconditioners for toeplitz-block least squares problems, SIAM Journal on Scientific and Statistical Computing 30 (1992) 1740–1768.
[38] T. F. Chan, J. A. Olkin, Circulant preconditioners for toeplitz-block matrices, Numer. Algor. 6 (1994) 89–101.
[39] S. P. Groth, A. G. Polimeridis, J. K. White, Accelerating the discrete dipole approximation via circulant precondi-

tioning, Journal of Quantitative Spectroscopy and Radiative Transfer 240 (2020) 106689.
[40] P. C. Chaumet, G. Maire, A. Sentenac, Accelerating the discrete dipole approximation by initializing with a scalar solution and using a circulant preconditioning, Journal of Quantitative Spectroscopy and Radiative Transfer 298 (2023) 108505.
[41] M. Yurkin, Handbook of Molecular Plasmonics, Chapter Computational Approaches for Plasmonics, Taylor & Francis Group, 2013.
[42] Y. Onoue, S. Fujino, N. Nakashima, Improved idr(s) method for gaining very accurate solutions, International Journal of Computer and Information Engineering 3 (2009) 1806 – 1811.