Preconditioning the discrete dipole approximation

Samuel P. Groth\textsuperscript{a*†}, Athanasios G. Polimeridis\textsuperscript{b}, and Jacob K. White\textsuperscript{a}

\textsuperscript{a}Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, USA.

\textsuperscript{b}Q Bio, Redwood, CA 94063, USA.

March 26, 2019

Abstract

The discrete dipole approximation (DDA) is a popular numerical method for calculating the scattering properties of atmospheric ice crystals. The standard DDA formulation involves the uniform discretization of the underlying volume integral equation, leading to a linear system with a block-Toeplitz Toeplitz-block matrix. This structure permits a matrix-vector product to be performed with $O(n \log n)$ complexity via the fast-Fourier transform (FFT). Thus, in principle, the system can be solved rapidly using an iterative method. However, it is well known that the convergence of iterative methods becomes increasingly slow as the optical size and refractive index of the scattering obstacle are increased. In this paper, we present a preconditioning strategy based on the multi-level circulant preconditioner of Chan and Olkin \cite{Chan1994} and assess its performance for improving this rate of convergence. In particular, we approximate the system matrix by a block-circulant circulant-block matrix which can be inverted rapidly using the FFT. We present numerical results for scattering by hexagonal ice prisms demonstrating that this serves as an effective preconditioning strategy, reducing simulation times by orders of magnitude in many cases. A Matlab implementation of this work is freely available online.

Keywords: Volume integral equation, discrete dipole approximation, preconditioning, circulant, electromagnetic scattering

1 Introduction

Since the publication of Purcell and Pennypacker’s seminal paper in 1973 \cite{Purcell1973} and the subsequent work of, amongst others, Draine and Flatau \cite{Draine1999}, Goedecke and O’Brien \cite{Goedecke2001}, and Yurkin and Hoekstra \cite{Yurkin2008}, the discrete dipole approximation (DDA) has proven popular for electromagnetic scattering simulations. Application areas include dust particles \cite{Purcell1973, Draine1999}, biological tissues \cite{Goedecke2001, O'Brien2002}, optical tweezers \cite{Yurkin2008}, and atmospheric ice crystals \cite{Yurkin2008}.

\textsuperscript{*}Present address: Department of Engineering, University of Cambridge, United Kingdom.

\textsuperscript{†}Corresponding author. Email address: samuelpgroth@gmail.com
The success of DDA is in no small part due to the fact that, when the underlying volume integral equation is discretized over a uniform (“voxelized”) grid, the system matrix obtains a block-Toeplitz-Toeplitz-block (BTTB) structure. This permits a matrix-vector product to be performed in $O(n \log n)$ operations via the fast-Fourier transform (FFT), where $n$ is the number of voxels in the grid \[10, 11\]. Therefore, the cost of solving the linear system via an iterative method is $O(n_{\text{iter}} n \log n)$, where $n_{\text{iter}}$ is the number of iterations required for convergence.

In principle, this modest growth of computational cost as $n$ increases should enable simulations for extremely large scattering obstacles to be efficiently performed. However, it is well known that, as the optical size and refractive index of the scatterer grow, $n_{\text{iter}}$ becomes prohibitively large (see, e.g., [4]). In fact, as we shall observe later, $n_{\text{iter}} \sim x^3$, where $x = ka$ is the size parameter of the particle, with $k$ the wavenumber and $a$ the radius of the obstacle’s smallest circumscribing sphere. In ice crystal simulations, for example, the scatterer may be up to a hundred wavelengths across for which $n_{\text{iter}}$ is so large as to make DDA infeasible. Therefore, an effective preconditioning strategy is required to temper the growth of $n_{\text{iter}}$.

This paper revisits the well-established circulant preconditioning techniques of Chan and Olkin [12, 13], in which the underlying Toeplitz matrix is approximated by a circulant, and applies it within the DDA context for the first time. This builds on work by the present authors where a similar approach was applied within a silicon photonics context in which the structures of interest typically have extreme length in one of the three physical dimensions [14, 15, 16].

To clarify briefly this approach, consider the structure of the DDA linear system. It is of the form

$$(D + T)x = b,$$  \hspace{1cm} (1)

where $D$ is a diagonal matrix containing the polarizabilities of the dipoles and $T$ is a BTTB matrix with three levels of Toeplitz structure, obtained from discretizing the dyadic Green’s function over the three-dimensional voxelized grid. For optically large scatterers, solving this system iteratively is expensive due to a large $n_{\text{iter}}$, hence we seek an appropriate preconditioner $P$ such that the modified system

$$P^{-1}(D + T)x = P^{-1}b$$  \hspace{1cm} (2)

is more efficiently solved, i.e., $n_{\text{iter}}$ is drastically reduced.

In order for $P$ to be effective, it must also be reasonably cheap to construct and invert. In our Toeplitz setting, a natural candidate for $P$ is a circulant matrix. Circulant matrices constitute a special class of Toeplitz matrix with the additional desirable property that they are diagonalized by the discrete Fourier transform, and hence can be inverted in $O(n \log n)$. Circulant preconditioners for Toeplitz systems have proven to be successful in many application areas (see [17] and the references therein), however, to the present authors’ knowledge, they have yet to be applied to DDA for practical EM scattering problems.

The shrewd reader would have observed that the matrix $D$ does not, in general, have a constant diagonal (only when the scatter is a homogeneous cuboid is this constant). Hence, $(D + T)$ does not inherit the Toeplitz property of $T$ for general scatterers. To circumvent this issue, we employ a simple averaging strategy to create a preconditioner of the form $P = \tilde{D} + C$, where $\tilde{D}$ has constant diagonal and is constructed by averaging the diagonal of $D$, and $C$ is a circulant approximation to $T$ à la Chan and Olkin [12]. Since $\tilde{D}$ has constant diagonal, then $\tilde{D} + C$ inherits the circulant nature of $C$, and hence the preconditioner $P$ can be cheaply inverted.
The example scatterers considered in this article are homogeneous hexagonal ice prisms of various size parameters and refractive indices. The discretized domain is the smallest box enclosing the scatterer (this is required for the FFT acceleration) so that the values along the diagonal of $D$ correspond to those of ice and air voxels. Here we have chosen to construct $D$ as $D = \text{mode} (\text{diag}(D)) I$, where $I$ is the identity matrix, i.e., the constant diagonal of $D$ is the modal average of the diagonal entries in $D$. For the hexagonal prisms considered here, this means that the preconditioner corresponds to the total bounding box “filled in” with ice. As we see in Section 4, this is an effective strategy for the ice crystal examples considered in this article. Improvements may potentially be made by considering either a different averaging technique or a more sophisticated “Toeplitz-plus-diagonal” preconditioner, as in [18], however these ideas are not explored here.

In terms of previous work on preconditioning for DDA, there appears to have been little development. Some brief experiments were presented in [19] where the simple point-Jacobi and Neumann polynomial preconditioners were used. However, large size parameters were not investigated and the improvement for small size parameters is extremely modest, if anything at all. More general preconditioning strategies exist, such as incomplete-LU, block-Jacobi [20], and the inverse fast multipole method [21], but these are potentially expensive, are not effective for high frequency problems, or are complicated to implement. The distinct advantages of circulant preconditioning are that it is well suited to the Toeplitz structure of DDA, is comparatively straightforward to implement, and it is inexpensive. Furthermore, as we present in Section 4, circulant preconditioning performs extremely well for the ice crystal scattering simulations considered here, providing speed-up factors of ten or more, and in many cases it enables previously inaccessible simulations to be performed on a desktop PC. Therefore, we believe that this paper presents the first viable preconditioning approach for an important class of DDA scattering simulations.

The layout of the paper is as follows. In Section 2, we provide details of the standard DDA formulation of [2] employed here. Also details of the Toeplitz structure of the system matrix are provided to facilitate the description of the circulant approximation in the following section. In Section 3, we review circulant preconditioning applied to Toeplitz matrices and its extension to Toeplitz-block matrices. In particular, we describe how the general approach of [12] is applied to our particular BTTB DDA matrix. Details of the algorithmic costings of assembling and applying the preconditioner are provided. We also present some pseudocode to help readers incorporate this preconditioning strategy into their own DDA codes. In Section 4, we consider the scattering of a polarized plane wave by hexagonal prisms of refractive indices $\mu = 1.2, 1.4, 1.6, 1.8, 2$ and of size parameters $x = 10, 20, 30, 40, 60, 80, 100$. We present the CPU times and iteration counts for the unpreconditioned and preconditioned iterative solves of the arising linear systems, using both GMRES and BiCG-Stab. For smaller size parameters, little gain is achieved, but for large size parameters, we observe acceleration factors of ten or more. In fact, for the largest size parameters, where unpreconditioned DDA fails to converge, we achieve convergence with the preconditioned DDA within an acceptable number of iterations. In Section 5, we provide some concluding remarks and ideas for the further development of the preconditioning strategy.

A Matlab implementation is openly available online (https://github.com/samuelpgroth/VoxScatter) which we hope will be useful to students and those wishing to develop this work further.
2 Integral equation formulation

The discrete dipole approximation, in its many forms, begins with the following integral equation representation for the time-harmonic ($e^{-i\omega t}$) electric field $E$ in the presence of a non-magnetic dielectric body $\Omega$:

$$E(r) = E^{\text{inc}}(r) + \int_{\Omega} G(r, r')\chi(r')E(r')dr',$$

(3)

where $E^{\text{inc}}$ is the incident field and $\chi(r) := (\varepsilon(r) - 1)/4\pi$ is the electric susceptibility, with $\varepsilon(r)$ the relative permittivity. The dyadic Green’s function, $G$, is defined as

$$G(r, r') = (k_0^2I + \nabla\nabla)e^{ik_0r}/r = e^{ik_0r}/r \left[ k_0^2(I_3 - \hat{r}\hat{r}^T) + ik_0 - \frac{1}{r^2}(I_3 - 3\hat{r}\hat{r}^T) \right],$$

(4)

where $r = |r - r'|$, $\hat{r} = (r' - r)/r \in \mathbb{R}^3 \times 1$, and $I$ is the $3 \times 3$ identity matrix [2,4]. Reordering (3) to obtain an integral equation for the unknown field gives

$$(I - G\chi)E = E^{\text{inc}},$$

(5)

where $G$ is the integral operator

$$Gf(r) = \int_{\Omega} G(r, r')f(r')dr'.$$

(6)

In the DDA of Purcell and Pennypacker [1], and Draine and Flatau [2], equation (5) is phrased in terms of the unknown polarization rather than the electric field. The polarization is defined as

$$P(r) = \chi(r)E(r)$$

(7)

and so the integral equation becomes

$$(\chi^{-1} - G)P = E^{\text{inc}},$$

(8)

for $\chi(r) \neq 0$ (of course $P(r) = 0$ where $\chi(r) = 0$ so we can neglect the contributions from such voxels). The formulation (8) is seen as desirable since there exists an accurate approximation for the self term via the Clausius-Mossotti relation. This enables the complicated evaluation of the singular portion of the integral to be sidestepped.

In this paper, we solve (8) via the “classical” DDA approach as expounded in, for example, [1,2]. A more rigorous approach would be to solve (8) via Galerkin’s method and evaluate the resulting double integrals with sophisticated numerical quadrature, as is done in [6] where it is used for magnetic resonance applications. Here we choose to present results for the simpler DDA approach, but point out that the preconditioning strategy presented can be applied to any volume integral equation scheme (e.g., DDA, Galerkin, collocation). The only requirement is that a cuboidal discretization grid is used, so that the resulting linear system has Toeplitz structure.
2.1 Discrete Dipole Approximation

DDA can be viewed as a collocation approach for solving equation [5] in which the singular self-term integrals are evaluated using semi-analytical means (namely, the Clausius-Mossotti relation) and the non-singular integrals are evaluated using the midpoint quadrature rule. We briefly summarize this approach.

Begin by writing the unknown polarization \( \mathbf{P} \) as

\[
\mathbf{P}(\mathbf{r}) = \sum_{j=1}^{N} \mathbf{c}_j \circ \mathbf{p}_j(\mathbf{r}),
\]

where each basis function \( \mathbf{p}_j \) is a three-dimensional unit pulse function supported on voxel \( j \) alone, i.e.,

\[
\mathbf{p}_j(\mathbf{r}) = \begin{cases} 
(1, 1, 1), & \text{if } \mathbf{r} \text{ in voxel } j, \\
0, & \text{otherwise},
\end{cases}
\]

\( \mathbf{c}_j = (c^{x}_j, c^{y}_j, c^{z}_j) \) are the unknown coefficient vectors, and \( \circ \) represents the Hadamard product. Upon substitution of the piecewise constant representation (9) into the integral equation (8), and then forcing this to be exact at the voxel centers, we obtain the linear system of equations

\[
\sum_{j=1}^{N} \mathbf{c}_j \circ \left\{ \chi^{-1}(\mathbf{r}_j)\mathbf{p}_j - \int_{\Omega_j} \mathbf{G}(\mathbf{r}_i, \mathbf{r}')\mathbf{p}_j(\mathbf{r}')d\mathbf{r}' \right\} = \mathbf{E}^{\text{inc}}(\mathbf{r}_i),
\]

for \( i = 1, \ldots, N \). This is the collocation approach for the solution of (8). We observe that when \( i = j \) (the self term), the integral in (10) is singular. In DDA schemes, this self term is given explicitly via the Clausius-Mossotti relation:

\[
\chi^{-1}(\mathbf{r}_i)\mathbf{p}_i - \int_{\Omega_i} \mathbf{G}(\mathbf{r}_i, \mathbf{r}')\mathbf{p}_i(\mathbf{r}')d\mathbf{r}' \approx \alpha_i^{-1}\mathbf{p}_i, \quad i = j,
\]

with \( \alpha_i \) the polarizability of a dipole at location \( \mathbf{r}_i \). Here we follow [2] and take \( \alpha = \alpha^{\text{LDR}} \), namely the lattice dispersion relation (LDR) correction to the Clausius-Mossotti polarizabilities \( \alpha^{\text{CM}} \). The definitions are given as

\[
\alpha_i^{\text{LDR}} = \frac{\alpha_i^{\text{CM}}}{1 + \alpha_i^{\text{CM}}(b_1 + m_1^2b_2 + m_1^2b_3S)(k_0\Delta)^2 - (2i/3)(k_0\Delta)^3},
\]

\[
\alpha_i^{\text{CM}} = \frac{3}{4\pi} \left( \frac{\epsilon_i - 1}{\epsilon_i + 2} \right),
\]

\( b_1 = -1.891531 \), \( b_2 = 0.1648469 \), \( b_3 = -1.770004 \), \( S := \sum_{j=1}^{3} (d_jE_j)^2 \),

where \( \epsilon_i = \mu_i^2 \) is the relative permittivity of the material occupying the \( i \)th voxel, and \( \mathbf{d}^i = (d_1, d_2, d_3) \) and \( \mathbf{E}_0 = (E_1, E_2, E_3) \) are unit vectors defining the direction and polarization of the incident field. Note that we use slightly different definitions of \( \alpha^{\text{CM}} \) and \( \alpha^{\text{LDR}} \) to that used in [2], namely we omit
the scaling by the voxel volume. But this difference is compensated for in our modified definition of the polarization $\mathbf{p}$, which we choose to fall in line with the more standard definition.

The non-singular integrals ($i \neq j$) are evaluated using the midpoint quadrature rule:

$$\int_{\Omega_j} \mathbf{G}(r_i, r') \mathbf{p}_j(r') \, dr' \approx \Delta^3 \mathbf{G}(r_i, r_j), \quad i \neq j,$$

where $\Delta$ is the voxel dimension (see Section 2.2). Such a crude quadrature scheme is accurate only for well-separated voxels. For nearby voxels, where the integral is close to singular, the midpoint rule is inaccurate. Schemes such as the digitized Green’s function [3], coupled dipole method [22], and the Galerkin implementation [6] use rigorous quadrature techniques to evaluate these integrals more accurately, and so are more accurate in general, and particularly for large permittivities. However, here we choose the more classical approach for simplicity. The important point is that both approaches are based on voxel discretizations, so can employ the preconditioning strategy proposed in this article.

2.2 Voxel discretization and Toeplitz structure

Although tetrahedral discretizations (e.g., [23]) can provide a more accurate geometrical representation, voxel discretizations have proven popular owing to the fact that they lead to a discrete system of convolution form, and hence permit a fast matrix-vector product via the FFT.

We begin the discretization by choosing an appropriate voxel dimension $\Delta$. Typically $\Delta$ is chosen so that $\lambda/(\mu \Delta) \geq 10$ in order to ensure an accurate approximation, where $\lambda$ is the wavelength of the incident field. In this article, we take $\lambda/(\mu \Delta) = 10$ to enable rapid simulations with meaningful results. Then a box bounding the scatterer is constructed, of dimension $l \Delta \times m \Delta \times n \Delta$ so that the voxel grid consists of $N = l \times m \times n$ voxels.

Discretizing the linear system of equations (10) over the voxel grid using (11) and (12), and using the ordering described in Algorithm 1 leads to a discrete system of the form

$$
\begin{bmatrix}
\alpha_x^{-1} & & \\
& \alpha_y^{-1} & \\
& & \alpha_z^{-1}
\end{bmatrix}
- \Delta^3
\begin{bmatrix}
\mathbf{G}_{xx} & \mathbf{G}_{xy} & \mathbf{G}_{xz} \\
\mathbf{G}_{xy} & \mathbf{G}_{yy} & \mathbf{G}_{yz} \\
\mathbf{G}_{xz} & \mathbf{G}_{yz} & \mathbf{G}_{zz}
\end{bmatrix}
\begin{bmatrix}
\mathbf{c}_x \\
\mathbf{c}_y \\
\mathbf{c}_z
\end{bmatrix}
= \begin{bmatrix}
\mathbf{E}_{x}^{\text{inc}} \\
\mathbf{E}_{y}^{\text{inc}} \\
\mathbf{E}_{z}^{\text{inc}}
\end{bmatrix}.
$$

The blocks $\alpha_x^{-1}$, $\alpha_y^{-1}$, $\alpha_z^{-1}$ are diagonal and each of the blocks $\mathbf{G}_{\alpha\beta}$ has BTTB structure on three levels, corresponding to the three physical dimensions of the problem. Note the symmetry in these blocks: only six of them are unique. Further, each of these blocks is either symmetric or anti-symmetric. This, combined with their BTTB structure, allows them to be each defined by a single row. Hence the storage cost for the $\mathbf{G}$ matrix is $O(6n)$. 

6
Algorithm 1 Computing the tensor $G \in \mathbb{C}^l \times m \times n \times 6$

1: $r_0 = (x(1), y(1), z(1))$ centerpoint of first voxel 
2: for $i = 1 : l$ do 
3: for $j = 1 : m$ do 
4: for $k = 1 : n$ do 
5: $r_c = (x(i), y(j), z(k))$ centerpoint of voxel 
6: $r = |r_c - r_0|$ 
7: if $r \neq 0$ then 
8: $\hat{r} = (r_c - r_0)/r$ 
9: $g = \frac{e^{ikr}}{r} \left[ k_0^2 (I_3 - T \hat{r}) + \frac{i k r - 1}{r} (I_3 - 3 T \hat{r}) \right]$ 
10: $G(i, j, k, 1) = g(1, 1), G(i, j, k, 2) = g(1, 2)$ 
11: $G(i, j, k, 3) = g(1, 3), G(i, j, k, 4) = g(2, 2)$ 
12: $G(i, j, k, 5) = g(2, 3), G(i, j, k, 6) = g(3, 3)$ 
13: end if 
14: end for 
15: end for 
16: end for

Further note that if the matrix $\alpha$ has a constant diagonal, i.e., the structure is homogeneous, then the matrix $\alpha^{-1} - G$ inherits the BTTB structure of $G$. This is the particular case in which circulant preconditioners prove most effective, as we discuss in the following section.

3 Circulant preconditioning

The circulant preconditioners employed here are based on those proposed in [12] for Toeplitz-block matrices, which are an extension of the optimal point-circulant preconditioners of [13]. We review here the salient features of multi-level circulant preconditioners and refer the reader to [12] for further details.

A Toeplitz matrix $T_n = \{t_{ij}\}_{i,j=0}^{l-1}$ is Toeplitz if $t_{ij} = t_{i-j}$, i.e., the diagonals are constant. Circulant matrices $C_l = [c_i]_{i,j=0}^{l-1}$ are also Toeplitz but with the additional property that every row of the matrix is a right cyclic shift of the row above, i.e, $c_{ij} = c_{(i-j) \mod l}$. Written out, these matrices have the respective forms

$$T_n = \begin{pmatrix} t_0 & t_{-1} & \cdots & t_{-(l-1)} \\ t_1 & t_0 & \cdots & \vdots \\ \vdots & \ddots & \ddots & t_{-1} \\ t_{l-1} & \cdots & t_1 & t_0 \end{pmatrix}, \quad C_n = \begin{pmatrix} c_0 & c_{l-1} & \cdots & c_1 \\ c_1 & c_0 & \cdots & \vdots \\ \vdots & \ddots & \ddots & c_{l-1} \\ c_{l-1} & \cdots & c_1 & c_0 \end{pmatrix}.$$ 

Note that circulant matrices have the desirable property that they are diagonalized by the discrete Fourier matrix $F_l$, such that $C_l = F_l^{-1} \Lambda_t F_l$, where $\Lambda_t = \text{diag}(F_l c)$ is a diagonal matrix with $c$ the defining column of $C_l$. Therefore, $C_l$ is inverted via the FFT in $O(l \log l)$ operations. For a Toeplitz matrix, T. Chan [13] proposed the optimal point-circulant preconditioner whose entries are given
by
\[
c_i = \begin{cases} 
\frac{t^{-i}t_i + i}{c_l}, & 0 \leq i \leq l - 1, \\
-c_{l+i}, & -(l-1) \leq i < 0. 
\end{cases}
\]  
(13)

This approximation is optimal in the sense that it is the closest circulant matrix to \( T_l \) in the Frobenius norm. There exist other circulant preconditioners (see, for example, the review [24]) and we anticipate the results presented in this paper would be similar if, for example, the Strang circulant preconditioner [25] were instead employed. We choose to employ T. Chan’s preconditioner since it is explicitly defined by the simple formula (13) and has been shown to be effective for many Toeplitz problems.

T. Chan’s preconditioner was extended to Toeplitz-block matrices in [12]. In our setting, the DDA matrix, \( G \), has \((3mn)^2\) Toeplitz blocks, each of size \( l \times l \). Let us denote such a matrix \( T_B \) (although we should keep in mind our DDA matrix). Then its circulant-block approximation, \( C_B \), is obtained by calculating the circulant approximation to each Toeplitz block via (13). These matrices are written as
\[
T_B = \begin{pmatrix}
T_{1,1} & T_{1,2} & \ldots & T_{1,3mn} \\
T_{2,1} & T_{2,2} & \ldots & T_{2,3mn} \\
\vdots & \vdots & \ddots & \vdots \\
T_{3mn,1} & T_{3mn,2} & \ldots & T_{3mn,3mn}
\end{pmatrix},
\]
and
\[
C_B = \begin{pmatrix}
C(T_{1,1}) & C(T_{1,2}) & \ldots & C(T_{1,3mn}) \\
C(T_{2,1}) & C(T_{2,2}) & \ldots & C(T_{2,3mn}) \\
\vdots & \vdots & \ddots & \vdots \\
C(T_{3mn,1}) & C(T_{3mn,2}) & \ldots & C(T_{3mn,3mn})
\end{pmatrix},
\]
where \( C(T) \) denotes the Chan circulant approximation, defined by (13), to \( T \). Having constructed \( C_B \), we then proceed to calculate its inverse via applications of the FFT. Each circulant block of \( C_B \) has the representation \( C(T_{ij}) = F^{-1}\Lambda_{ij}F \). Defining \( F = I \otimes F \), we then have that
\[
C_B = [C(T_{ij})]_{ij=1}^{3mn} = [F^{-1}\Lambda_{ij}F]_{ij=1}^{3mn} = F^{-1}[\Lambda_{ij}]_{ij=1}^{3mn}F.
\]  
(14)

The matrix \([\Lambda_{ij}]_{ij=1}^{3mn}\) is an \( 3lmn \times 3lmn \) diagonal-block matrix, where the diagonal blocks have size \( l \times l \). As described in [12], this matrix is easily collapsed to a block-diagonal matrix \( D \) after multiplication by a permutation matrix \( P \), where
\[
\text{diag}(D_1, \ldots, D_l) = P[\Lambda_{ij}]_{ij=1}^{3mn}P^T.
\]  
(15)

Therefore, the inverse of \( C_B \) is given by
\[
C_B^{-1} = F^{-1}P^T\text{diag}(D_1^{-1}, \ldots, D_l^{-1})PF.
\]  
(16)

We term \( C_B \) the 1-level circulant preconditioner and illustrate its construction in Figure [1]. The cost of the inversion of \( C_B \) is dominated by the inversion of the \( l \) dense blocks \( D_i \), each of size \( 3mn \times 3mn \). Therefore, the cost is \( O(l(3mn)^3) \).

If \( m \) and \( n \) are small, \( C_B \) can be a cheap preconditioner. If they are not small, one may resort to a second level of circulant approximation, applied this time to each of the dense blocks \( D_i \). In our
1-level circulant approximation

\[
\begin{bmatrix}
\end{bmatrix}
\]

Diagonalize with FFT

\[
\begin{bmatrix}
\end{bmatrix}
\]

Permute

\[
\begin{bmatrix}
\end{bmatrix}
\]

Figure 1: **The 1-level circulant approximation.** Once the circulant-block approximation is constructed, the blocks are diagonalized with the FFT to create a diagonal-block matrix, then this is permuted to obtain a block-diagonal matrix to be inverted. In practice, it is not necessary to perform the first two steps and one can create the final block-diagonal matrix directly. Note that in our problem, where we have three levels of Toeplitz structure, the blocks in the final matrix are themselves BTTB with two levels of Toeplitz structure. This allows the process to be repeated for each of the blocks thus creating a further level of circulant approximation.

In the BTTB case, the blocks \( D_i \) are themselves Toeplitz-block, thus allowing the above procedure to be repeated for each \( D_i \), leading to a **2-level circulant preconditioner** which we denote by \( C_{B_2} \). The matrix \( C_{B_2} \) can be written as

\[
C_{B_2} = F^{-1}P^T \text{diag}(C_{B_1}(D_1), \ldots, C_{B_1}(D_l))PF,
\]

where \( C_{B_1}(D_i) \) denotes the 1-level circulant approximation

\[
C_{B_1}(D_i) = F^{-1}P^T \text{diag}(\overline{D}_1, \ldots, \overline{D}_m)PF,
\]

where \( \overline{D}_i \) are new blocks, of size \( 3n \times 3n \). The lines above \( F \) and \( P \) are to highlight that they are of the dimension appropriate for \( \overline{D}_i \). An illustration of the 2-level circulant approximation is shown in Figure 2. The resulting block-diagonal matrix has \( lm \) blocks, each of size \( 3n \times 3n \), which must inverted to obtain \( C_{B_2}^{-1} \) for use as our preconditioner:

\[
C_{B_2}^{-1} = F^{-1}P^T \text{diag}(C_{B_1}^{-1}(D_1), \ldots, C_{B_1}^{-1}(D_l))PF.
\]

Again, the inversion cost is dominated by the inversion of the \( lm \) blocks and so is now \( O(lm(3n^3)) \).

### 3.1 Algorithms

We present a few details as to the practical construction and inversion of the 2-level circulant preconditioner \( C_{B_2} \). First we remind the reader that matrix we are wishing to approximate is of the form

\[
\alpha^{-1} - \Delta^3 G
\]

where \( \alpha \) is a diagonal matrix with each entry being the polarizability of the appropriate voxel and \( G \) is our BTTB DDA matrix. In general, \( \alpha \) does not have a constant diagonal unless we are dealing
with a homogeneous cuboid. In the examples considered in Section 4 we deal with homogeneous hexagonal prisms so that the polarizabilities take one of two values, that of the “ice” voxels or of the “air” voxels.

The endeavour is to create the 2-level circulant approximation $C_{B_2}$ with the hope that $C_{B_2} \approx \alpha^{-1} - \Delta^3 G$ so that it acts as a good preconditioner. To do this, we first construct the 2-level circulant approximation to the BTTB matrix $G$ and denote this $\tilde{C}_{B_2}$. Then we must construct a constant diagonal matrix that approximates $\alpha$, denoting this $\tilde{\alpha}I$. Now we have that the matrix

$$C_{B_2} := \tilde{\alpha}^{-1}I - \Delta^3 \tilde{C}_{B_2} \tag{20}$$

is also circulant and hence appropriate as our circulant preconditioner. We stress the importance of the construction of the diagonal matrix $\tilde{\alpha}I$ since, when $\alpha$ is not a constant diagonal (i.e., when the scatterer is not a homogeneous cuboid), the matrix $\alpha^{-1} - \Delta^3 C_{B_2}$ does not inherit the circulant properties of $C_{B_2}$ and hence is not cheaply inverted. The choice made here is to take $\tilde{\alpha}$ as the value of the “ice” voxels, however it may be the case that some $\tilde{\alpha}$ derived from averaging over the $\alpha_i$ leads to superior performance (as was seen in [16] for the 1-level circulant preconditoner). But we do not explore this question in this article. The final step is the inversion of $C_{B_2}$, which is performed in a parallel loop over its $ln$ diagonal blocks, each of dimension $3n$.

Now we state a few details about the efficient construction of the circulant approximation to $G$. This construction can be performed efficiently by exploiting the symmetries within the constituent blocks of $G$. Each of the six unique blocks of $G$ has a three-level Toeplitz structure and on each of these levels the elements blocs are arranged either symmetrically or anti-symmetrically - these symmetries are provided in Table 1.

As an example, let us take the block $G_{xz}$. First we wish to calculate the 1-level circulant approximation to this block in the $x$-direction. We know that this block is anti-symmetric on the first level, so its circulant approximation is given as shown in Algorithm 2; note the minus sign in this version of the circulant approximation [13]. After having performed this circulant approximation for each
Table 1: Symmetry (+) and anti-symmetry (−) patterns within the different levels of the BTTB structure of the constituent blocks of the matrix \( G \). This knowledge is required in the efficient construction of the circulant preconditioner.

|        | 1st level \((x)\) | 2nd level \((y)\) | 3rd level \((z)\) |
|--------|------------------|------------------|------------------|
| \( G_{xx} \) | + | + | + |
| \( G_{xy} \) | − | − | + |
| \( G_{xz} \) | − | + | − |
| \( G_{yy} \) | + | + | + |
| \( G_{yz} \) | + | − | + |
| \( G_{zz} \) | + | + | + |

Algorithm 2 Compute 1-level circulant approximation to \( G_{xz} \)

1. \( G_{xz} = G(:,:,;3) \)
2. \( c(1,:,:) = G_{xz}(1,:,:) \) the diagonal entries of the circulant and the Toeplitz matrices equal
3. for \( i = 1 : l \) do
4. \( c(i,:,:) = -\frac{i+1-i}{l} G_{xz}(i,:,:) + \frac{i-1}{l} G_{xz}(l-i+2,:,:) \) note the minus sign due to antisymmetry
5. end for
6. \( \tilde{C}_{xz}^{(1)} = \text{FFT}(c) \) provides the Fourier transform of columns of \( c \)

Algorithm 3 Compute 2-level circulant approximation to \( G_{xz} \)

1. for \( i = 1 : l \) do
2. \( \tilde{C}_{xz}^{(1)}(i,:,:) = \tilde{C}_{xz}^{(1)}(i,:,:) \)
3. \( c(1,:) = \tilde{C}_{xz}^{(1)}(1,:) \) the diagonal entries of the circulant and the Toeplitz matrices equal
4. for \( j = 1 : m \) do
5. \( c(j,:) = \frac{m+1-i}{m} \tilde{C}_{xz}^{(1)}(i,j,:) + \frac{i-1}{l} \tilde{C}_{xz}^{(1)}(i,m-j+2,:) \)
6. end for
7. \( \tilde{C}_{xz}^{(2)}(i,:,:) = \text{FFT}(c) \) provides the Fourier transform of columns of \( c \)
8. end for

Algorithm 2 and Algorithm 3 serve to illustrate that the generation of the 1- and 2-level circulant approximations are fairly straightforward and can be performed directly from the defining tensor \( G \in \mathbb{C}^{l \times m \times n \times 6} \) given in Algorithm 1, which is constructed within all FFT-accelerated DDA
implementations.

3.2 Costings

Costings were provided in Chan and Olkin [12] but for a general Toeplitz-block matrix. Here we provide the relevant costings for our symmetric system. Following [12] we consider a floating-point operation (flop) as one multiplication plus one addition. We also denote the cost of applying the FFT to a vector of length $n$ as $\text{fft}(n)$, which is typically $5\log_2(n)$ flops for standard FFT algorithms such as FFTW [26].

First we consider the setup (including the inversion) and per-iteration application costs of the 1-level circulant preconditioner.

1-level: setup

1. Point-circulant approximation via (13) of the $6mn$ unique blocks of length $l$: $6lmn$ flops
2. $6mn$ FFTs of length $l$ to generate $\tilde{C}^{(1)}$: $6mn \cdot \text{fft}(l)$
3. Generate $l$ diagonal blocks from $\tilde{C}^{(1)}$ using knowledge of BTTB structure and symmetries in Table 1 \sim free.
4. Inversion of the $l$ dense diagonal blocks of size $(3mn) \times (3mn)$: $\frac{1}{4}l(3mn)^3$.

$$1\text{-level setup cost} = \frac{1}{3}l(3mn)^3 + 6lmn + 6mn \cdot \text{fft}(l).$$

1-level: application (per iteration)

1. $3mn$ FFTs, each of length $l$: $3mn \cdot \text{fft}(l)$
2. Multiplication with the $l$ diagonal blocks $D_i$ of size $(3mn) \times (3mn)$: $l(3mn)^2$.
3. $3mn$ inverse FFTs, each of length $l$: $3mn \cdot \text{fft}(l)$

$$1\text{-level application cost} = l(3mn)^2 + 6mn \cdot \text{fft}(l).$$

Next we consider the setup and per-iteration application costs of the 2-level circulant preconditioner.

2-level setup

1. Steps 1 and 2 from the 1-level setup to generate the $l$ diagonal blocks: $6lmn + 6mn \cdot \text{fft}(l)$
2. Point-circulant approximation via (13) of the $6ln$ blocks of length $m$: $6lmn$.
3. $l \times 6n$ FFTs of length $m$ to generate $\tilde{C}^{(1)}$: $6ln \cdot \text{fft}(m)$
4. Generate \( lm \) diagonal blocks from \( \tilde{C}^{(2)} \) using knowledge of BTTB structure and symmetries in Table 1 \( \sim \) free.

5. Inversion of each of the \( lm \) dense diagonal blocks of size \( (3n) \times (3n) \): \( \frac{1}{3} l m (n)^3 \).

\[
\text{2-level setup cost} = \frac{1}{3} l m (3n)^3 + l (6mn + 6n \cdot \text{fft}(m)) \\
+ 6lmn + 6mn \cdot \text{fft}(l).
\]

2-level application (per iteration)

1. \( 3mn \) FFTs of length \( l \): \( 3mn \cdot \text{fft}(l) \)

2. For each of the \( l \) blocks:
   (i) \( 3n \) FFTs of length \( m \): \( 3n \cdot \text{fft}(m) \)
   (ii) Multiplication with the \( m \) blocks of size \( (3n) \times (3n) \): \( m(3n)^2 \)
   (iii) \( 3n \) inverse FFTs of length \( m \): \( 3n \cdot \text{fft}(m) \)

3. \( 3mn \) inverse FFTs of length \( l \): \( 3mn \cdot \text{fft}(l) \).

\[
\text{2-level application cost} = lm(3n)^2 + 6mn \cdot \text{fft}(l) + 6ln \cdot \text{fft}(m).
\]

The setup cost of the 1-level preconditioner is dominated by the block inversion and so the complexity is \( O(lm^3n^3) \). For problems in which \( m, n \ll l \), this cost is low and hence this preconditioner is feasible - this was seen for silicon photonics geometries in [16]. However, for ice crystal applications the geometries of interest are typically optically large in all three dimensions. For example, a cube with ten wavelengths across (a size parameter of \( \sim 31 \)) discretized at a resolution of 10 voxels per wavelength would require the storage and inversion of 100 dense matrix blocks of dimension \( 3 \times 10^4 \), a cost that is extremely demanding of most computers. Switching instead to the 2-level preconditioner, for this example, requires the storage and inversion of \( 10^4 \) dense matrix blocks of dimension \( 3 \times 10^2 \), a much more manageable task. Furthermore, this task can be performed in parallel and hence very rapidly, as we shall see in Section 4. For this reason of cost, we shall be applying only the 2-level preconditioner in this paper. For details on the performance of the 1-level, the reader is referred to [16]. We also note that a 3-level circulant approximation is possible, however it was found in a preliminary study to yield poorer results so we do not consider it here.

In terms of size parameter \( x \), since \( l, m, n \sim x \), we summarize the costings in Table 2. For reference, we also provide the costings for assembling the defining tensor of \( G \) and performing an MVP with it. We observe from the table that the cost of the preconditioner is greater than that of the integral operator, however not substantially so. Furthermore, the constants are hidden. In Section 4 we provide timings and memory consumption figures in order to observe these costs in practice.
### Preconditioner

|                  | Storage cost (memory) | Setup cost (time) | MVP (time) |
|------------------|-----------------------|-------------------|------------|
|                  | $O(x^4)$              | $O(x^5)$          | $O(x^4)$  |

### Operator

|                  | Storage cost (memory) | Setup cost (time) | MVP (time) |
|------------------|-----------------------|-------------------|------------|
|                  | $O(x^4)$              | $O(x^5)$          | $O(x^4 \log_2 x)$ |

Table 2: Costings for the preconditioner and integral operator in terms of size parameter. These are derived from expressions given in the text body by assuming $l, m, n \sim x$.

### 4 Numerical results

In order to test the performance of the circulant preconditioner for a realistic application, we consider the scattering of a plane wave by hexagonal plates with a variety of refractive indices and size parameters. In particular, we consider the scattering setup shown in Figure 3. The incident wave is polarized in the $z$-direction and travels in the positive $x$-direction, i.e., it has the form $E^{\text{inc}} = (0, 0, 1)e^{ikx}$. We consider two different values for the aspect ratio $L/a = 0.1, 0.2$, where $L$ is the height of the plate and $a$ is the radius of the smallest circumscribing circle of the hexagonal face. The refractive indices considered are $\mu = 1.2, 1.4, 1.6, 1.8, 2$ and the size parameters are $x = 10, 20, 30, 40, 60, 80, 100$, where $x = ka$. These parameter values are chosen to allow for a soft comparison to [4] where iteration counts are given for DDA, albeit there for scattering by spheres.

We present performance results for the iterative solves of the linear system using both the generalized minimum residual method (GMRES) and the biconjugate gradient stabilized method (BiCG-Stab) on an Intel (R) Xeon (R) CPU E5-2680 v4 @ 2.40GHz machine. BiCG-Stab is a popular iterative solver for DDA since it is fast, however its convergence is not guaranteed. GMRES on the other hand is slower and more memory intensive owing to the storage and use of the Krylov vectors, but its convergence is guaranteed if the entire Krylov subspace is kept, which we do here. We note that using restarted GMRES may lead to superior performance but we do not explore that in this article. As a stopping tolerance for the iterative solvers, we use $10^{-5}$, following [4].
\begin{table}
\centering
\begin{tabular}{|c|c|c|c|c|c|c|c|c|c|c|}
\hline
\(\mu\) & \(x\) & \#Total & & & & & & & & \\
 & & voxels & \# & Unpreconditioned & & & & Preconditioned & \\
 & & & & & & & & & \\
 & & & & & GMRES & BiCG-Stab & GMRES & BiCG-Stab & \\
 & & & & & Its. & Solve(s) & Its. & Solve(s) & \\
 & & & & & Build(s) & & & \\
1.2 & 10 & \(2.7 \times 10^3\) & 8 & 0.38 & 8 & 0.10 & 0.29 & 6 & 0.39 & 7 & 0.08 \\
20 & 2.1 \times 10^4 & 14 & 1.9 & 15 & 0.36 & 0.99 & 11 & 2.0 & 12 & 0.45 \\
30 & 6.9 \times 10^4 & 28 & 7.3 & 33 & 1.9 & 2.4 & 27 & 8.0 & 34 & 3.1 \\
40 & 1.6 \times 10^5 & 58 & 28 & 64 & 9.7 & 4.9 & 31 & 21 & 34 & 8.6 \\
60 & 5.5 \times 10^5 & 189 & 72 & 247 & 120 & 17 & 32 & 29 & 36 & 31 \\
80 & 1.2 \times 10^6 & 416 & 3,800 & 556 & 620 & 40 & 37 & 184 & 41 & 86 \\
100 & 2.4 \times 10^6 & 795 & 36,000 & 1062 & 2400 & 93 & 42 & 387 & 47 & 200 \\
1.4 & 10 & \(3.5 \times 10^4\) & 10 & 0.45 & 11 & 0.071 & 0.63 & 7 & 0.47 & 8 & 0.072 \\
20 & 3.5 \times 10^4 & 41 & 4.6 & 47 & 1.4 & 1.6 & 27 & 4.2 & 39 & 1.9 \\
30 & 1.1 \times 10^5 & 158 & 61 & 233 & 20 & 3.3 & 35 & 15 & 43 & 6.1 \\
40 & 2.5 \times 10^5 & 400 & 670 & 554 & 120 & 7.2 & 42 & 39 & 52 & 19 \\
60 & 8.1 \times 10^5 & 1533 & 27,000 & 2957 & 2100 & 24 & 64 & 180 & 98 & 120 \\
80 & 1.9 \times 10^6 & & & & & & & & & \\
100 & 3.8 \times 10^6 & & & & & & & & & \\
1.6 & 10 & \(6.7 \times 10^4\) & 16 & 0.75 & 17 & 0.13 & 0.43 & 11 & 0.76 & 12 & 0.15 \\
20 & 4.5 \times 10^4 & 85 & 11 & 111 & 4.1 & 1.6 & 29 & 5.6 & 33 & 2.2 \\
30 & 1.6 \times 10^5 & 573 & 850 & 1124 & 140 & 4.6 & 52 & 30 & 105 & 23 \\
40 & 3.6 \times 10^5 & 1339 & 9500 & 3460 & 1100 & 10 & 73 & 92 & 132 & 71 \\
60 & 1.2 \times 10^6 & & & & & & & & & \\
80 & 2.9 \times 10^6 & & & & & & & & & \\
100 & 5.9 \times 10^6 & & & & & & & & & \\
1.8 & 10 & \(8.7 \times 10^4\) & 18 & 0.50 & 20 & 0.18 & 0.52 & 12 & 0.51 & 13 & 0.21 \\
20 & 6.9 \times 10^4 & 313 & 120 & 601 & 32 & 2.3 & 36 & 6.5 & 49 & 4.5 \\
30 & 2.3 \times 10^5 & 1,224 & 5,100 & 3,471 & 730 & 6.4 & 72 & 52 & 425 & 160 \\
40 & 5.5 \times 10^5 & & & & & & & & & \\
60 & 1.7 \times 10^6 & & & & & & & & & \\
80 & 4.2 \times 10^6 & & & & & & & & & \\
100 & 8.2 \times 10^6 & & & & & & & & & \\
2 & 10 & \(1.1 \times 10^4\) & 20 & 0.56 & 24 & 0.24 & 0.61 & 13 & 0.57 & 13 & 0.25 \\
20 & 8.5 \times 10^4 & 502 & 350 & 1,313 & 91 & 2.7 & 46 & 11 & 139 & 19 \\
30 & 3.2 \times 10^5 & & & & & & & & & \\
40 & 7.3 \times 10^5 & & & & & & & & & \\
60 & 2.4 \times 10^6 & & & & & & & & & \\
80 & 5.9 \times 10^6 & & & & & & & & & \\
100 & 1.1 \times 10^7 & & & & & & & & & \\
\hline
\end{tabular}
\caption{Table 3: Aspect ratio \(L/a = 0.1\). The symbol \(\bullet\) represents no convergence for GMRES within 2000 iterations and BiCG-Stab within 4000 iterations. The symbol \(\circ\) signifies that the computer’s memory limit was exceeded by GMRES’s Krylov subspace.}
\end{table}

Table 3 shows the iteration counts and timings for the hexagonal plate of aspect ratio \(L/a = 0.1\). We are employing GMRES and BiCG-Stab as the iterative solvers and choose to cease the solves.
Figure 4: Iteration counts for GMRES (tol = $10^{-5}$) for the hexagonal plate with refractive index $\mu = 1.2$, aspect ratio $L/a = 0.1$, and size parameter ranging from 10 to 100. Without preconditioning, the iteration count grows slightly slower than $O(x^3)$ whereas with the preconditioner, the growth is approximately $O(x^{1/2})$.

Figure 5: Convergence of BiCG-Stab and GMRES for a hexagonal plate of aspect ratio $L/a = 0.1$, refractive index $\mu = 1.2$, and size parameter $x = 100$. For this example, there are $1.8 \times 10^6$ unknowns. The preconditioner took 92.6s to build and the solve times were the following: unpreconditioned BiCG-Stab - 1062s; preconditioned BiCG-Stab - 196s; unpreconditioned GMRES - 36,000s; preconditioned GMRES - 387s.

If convergence is not achieved within 2000 and 4000 iterations, respectively. One can observe that, with no preconditioning, the iteration count grows approximately as $O(x^3)$ with both GMRES and BiCG-Stab, so that for large values of $\mu$, DDA simulations are infeasible, which motivates the use of a good preconditioner. We note further that BiCG-Stab is indeed faster than GMRES for many of the unpreconditioned simulations.

With the preconditioner, the iteration count grows much more slowly as the size parameter increases. For $\mu = 1.2$, the iteration count growth is $O(x^{1/2})$ (as shown in Figure 4) whereas for larger $\mu$, the growth is closer to $O(x)$, thereby permitting much larger size parameter simulations compared to without preconditioning. It is worthwhile to observe that BiCG-Stab yields faster preconditioned
simulations than GMRES for $\mu = 1.2, 1.4$ but for higher values of the refractive index $\mu$, GMRES proves more reliable. However, for $\mu = 1.6, 1.8$ we see that the memory of the machine is exceeded by the Krylov subspace generated by GMRES at the largest size parameters. This suggests that exploring the use of GMRES with restarts would be worthwhile from a performance perspective. In any case, we observe that the preconditioner is providing an excellent improvement in the performance of iterative solvers. To illustrate this further, in Figure 5 we present the convergence of the relative residual of GMRES and BiCG-Stab for $\mu = 1.2, x = 100, L/a = 0.1$.

In terms of timings, for the small size parameters, where little gain is achieved using the preconditioner, the simulation times are comparable between preconditioned and unpreconditioned solves, and the overhead of building the preconditioner is less than a second in all cases. For the larger size parameters, we observe the huge advantage gained by employing the preconditioner. For example, for $\mu = 1.4, x = 60$, the solve with BiCG-Stab without a preconditioner takes 35 minutes, whereas with the preconditioner the solve takes 2.5 minutes (including the 24s preconditioner build time) – a factor of 14 speed up. So the small overhead time required to build the preconditioner is certainly worth it.

In Figure 6 we present more details of the overhead required to use the preconditioner. Figure 6(a) compares the assembly times of the preconditioner and the integral operator $G$ for growing size parameter $x$. In terms of the assembly, the time for $G$ grows as $\mathcal{O}(x^3)$ as can be seen from Algorithm 1. The cost of assembling the preconditioner is slightly higher and appears to increase as $\mathcal{O}(x^3)$ also, which is contrary to our prediction of $\mathcal{O}(x^5)$ in Section 3.2. This is likely due to the fact that the assembly is parallelized and Matlab’s matrix inversion routines are extremely efficient and so it takes a very large $x$ before the asymptotic range of $\mathcal{O}(x^5)$ is reached. In this range of $x$, the preconditioner takes approximately 2.5 times as long to assemble as the operator $G$ but we note that the total simulation time is dominated by the iterative solve, so this increased setup time is worth it as long as the iteration count is reduced sufficiently.
| $\mu$ | $x$ | #Total voxels | Unpreconditioned | Preconditioned |
|------|-----|---------------|------------------|---------------|
|      |     |               | GMRES | BiCG-Stab | GMRES | BiCG-Stab |
|      |     |               | Its. | Solve | Its. | Solve | Its. | Solve | Its. | Solve |
| 1.2  | 10  | $5.3 \times 10^3$ | 11 | 0.49 | 11 | 0.08 | 0.37 | 9 | 0.29 | 9 | 0.10 |
| 20   | 4.1 | $10^4$         | 27 | 2.5  | 29 | 1.1  | 1.3  | 23 | 2.80 | 32 | 1.93 |
| 30   | 1.4 | $10^5$         | 65 | 19   | 76 | 9.0  | 3.8  | 29 | 10.7 | 36 | 6.66 |
| 40   | 3.1 | $10^5$         | 120| 100  | 149| 37   | 8.4  | 35 | 30.3 | 38 | 16.8 |
| 60   | 1.1 | $10^6$         | 308| 1700 | 391| 370  | 40   | 41 | 136  | 46 | 91.8 |
| 80   | 2.5 | $10^6$         | 707| 20,000| 906| 2,100| 130  | 55 | 610  | 72 | 630 |
| 100  | 4.8 | $10^6$         | o  | o    | o  | o    | 290  | 61 | 2,500| 79 | 2,800 |
| 1.4  | 10  | $8.8 \times 10^3$ | 22 | 0.48 | 25 | 0.20 | 0.60 | 17 | 0.58 | 21 | 0.30 |
| 20   | 6.3 | $10^4$         | 116| 20   | 156| 7.6  | 1.9  | 31 | 5.1  | 46 | 3.4 |
| 30   | 2.0 | $10^5$         | 393| 510  | 563| 100  | 5.8  | 45 | 27   | 63 | 19 |
| 40   | 5.0 | $10^5$         | 852| 5,400| 1,692| 760| 17 | 60 | 89   | 96 | 82 |
| 60   | 1.7 | $10^6$         | •  | •    | •  | •    | 74   | 95 | 680  | 159| 670 |
| 80   | 4.0 | $10^6$         | •  | •    | •  | •    | 230  | 149| 6,100| 388| 8,500 |
| 100  | 7.7 | $10^6$         | •  | •    | •  | •    | 520  | o  | o    | 956| 19,000 |
| 1.6  | 10  | $1.1 \times 10^4$ | 34 | 0.80 | 41 | 0.40 | 0.50 | 20 | 0.76 | 26 | 0.43 |
| 20   | 9.0 | $10^4$         | 384| 230  | 641| 44   | 2.5  | 42 | 9.7  | 58 | 6.8 |
| 30   | 3.1 | $10^5$         | 1,148| 6,000| 2,730| 670| 9.4 | 81 | 74   | 236| 110 |
| 40   | 7.2 | $10^5$         | •  | •    | •  | •    | 26   | 114| 300  | 330| 411 |
| 60   | 2.5 | $10^6$         | •  | •    | •  | •    | 130  | 200| 4,400| 3,054| 23,000 |
| 80   | 5.9 | $10^6$         | •  | •    | •  | •    | 362  | o  | o    | •  | • |
| 1.8  | 10  | $1.7 \times 10^4$ | 79 | 3.2  | 131| 1.8  | 0.93 | 25 | 1.6  | 33 | 0.92 |
| 20   | 1.4 | $10^5$         | 802| 1,400| 2,333| 230| 3.9 | 61 | 23   | 178| 31 |
| 30   | 4.4 | $10^5$         | •  | •    | •  | •    | 14   | 127| 205  | •  | • |
| 40   | 1.1 | $10^6$         | •  | •    | •  | •    | 41   | 222| 1,300| •  | • |
| 60   | 3.5 | $10^6$         | •  | •    | •  | •    | 190  | 461| 20,000| •  | • |
| 2   | 10  | $2.1 \times 10^4$ | 133| 9.1  | 256| 4.2  | 0.78 | 30 | 1.8  | 42 | 1.2 |
| 20   | 1.8 | $10^5$         | 1,268| 4,400| •  | •    | 5.3  | 111| 75   | •  | • |
| 30   | 6.0 | $10^5$         | •  | •    | •  | •    | 23   | 216| 620  | •  | • |
| 40   | 1.5 | $10^6$         | •  | •    | •  | •    | 65   | 443| 5,740| •  | • |
| 60   | 4.8 | $10^6$         | •  | •    | •  | •    | 280  | o  | o    | •  | • |

Table 4: Aspect ratio $L/a = 0.2$. The symbol • represents no convergence for GMRES within 2000 iterations and BiCG-Stab within 4000 iterations. The symbol o signifies that the computer’s memory limit was exceeded by GMRES’s Krylov subspace.

Also in Fig. 6(a) are shown the times required to perform matrix-vector products with the preconditioner as well as with the operator $G$. We observe that they are comparable and agree well with the costings provided in Section 3.2. Since they are comparable, this suggests that the break even point in using the preconditioner is to reduce the iteration count to approximately half. That is, if the iteration count for the preconditioned solve is smaller than half that of the unpreconditioned solve, then employing the preconditioner is worthwhile. Indeed, we see in Table 3 that this is indeed the case for the majority of the parameter combinations.
Finally, in Fig. 6(b) we compare the memory required to store the preconditioner and integral operator. The memory required to store the preconditioner grows as $O(x^4)$ compared to the $O(x^3)$ required for the operator, with the preconditioner being more expensive for $x \geq 30$. This increased memory consumption is not problematic for the problems looked at here, however for higher resolution simulations and/or larger scatterers, this may prove a bottleneck. It is likely that some compression of the preconditioner is possible, as was seen in [16] for the 1-level circulant preconditioner.

The final results presented are for the same scattering setup but now with an aspect ratio of $L/a = 0.2$, in Table 4. This scatterer is twice as large as the previous one so we would expect that the iteration counts for the unpreconditioned solves are even higher. Indeed we find this is the case - unpreconditioned solves require roughly twice the number of iterations compared to the $L/a = 0.1$ example. The preconditioned solves also require more iterations but the increase is slightly less severe. The most noteworthy aspect is that now the timings for the preconditioned solves with GMRES are more comparable to those with BiCG-Stab, and certainly more reliable. For $\mu = 1.8, 2$, BiCG-Stab struggles to converge except for at the lowest values of $x$. Again we see that for the most challenging problems, the memory consumption of GMRES without restarts is prohibitive, again motivating future numerical experimentation with restarted GMRES.

5 Conclusions and future work

In this paper, we have presented the first application of a multi-level circulant preconditioner to electromagnetic scattering simulations with the discrete dipole approximation. Indeed, we believe that this is the first presentation of a viable preconditioner of any kind for ice crystal simulations within the DDA literature.

In particular, we applied the so-called “optimal” multi-level preconditioner of Chan and Olkin [12] in the simulation of scattering by homogeneous hexagonal plates of various size parameters and refractive indices. Via a consideration of the symmetrical block-Toeplitz Toeplitz-block structure of the voxel-discretized dyadic Green’s function $G$, we provided costings for the assembly and application of the 2-level circulant preconditioner. These costings suggest that the number of flops required for the assembly of the preconditioner scales with the size parameter $x$ as $O(x^5)$, compared to the standard DDA cost of $O(x^3)$. However, it was seen in the numerical experiments that the cost scaling of the preconditioner is milder than this, close to $O(x^3)$ for the size parameters considered. So that the preconditioner is approximately 2.5 times more expensive than the assembly of $G$, which is of the order of seconds or minutes.

Further, we observed that the cost of applying the preconditioner is almost the same as the cost of an MVP with $G$, suggesting that a reduction in iteration count by a factor of two is the break-even point. For the vast majority of the parameter combinations considered, a reduction factor far greater than two was achieved. In fact, the iteration count appears to grow as $O(x)$ (or even more mildly) compared to the $O(x^3)$ growth seen for unpreconditioned solves. Hence, for larger size parameters, the reduction in solve time achieved by using the preconditioner is greatest. For some parameter combinations, the preconditioned solves were up to 15 times faster. More remarkably, for large size parameter and large refractive index scatterers, the preconditioner enables previously
infeasible problems to become tractable, thereby enabling a wider applicability of DDA.

This work therefore has shown that circulant preconditioners for DDA simulations can be extremely effective. The scatterers considered here, although limited in their variety, are already of importance in atmospheric physics applications. For more general scattering setups, further experimentation and, potentially, development is required, however it seems probable that a circulant preconditioning strategy will prove helpful. We conclude by discussing some directions for future development of this work.

Recall that in Section 3.1 a choice was made in the construction of the 2-level preconditioner. In particular, we created a constant diagonal matrix $\tilde{\alpha}^{-1}I$ to replace the matrix $\alpha^{-1}$. This was done by simply using the value of $\alpha$ for “ice” for the air voxels also. Such an approximation step, however, may not be necessary. Instead, investigation of “Circulant-plus-diagonal” preconditioners, such as in [18], may prove fruitful. Or potentially a more sophisticated point-circulant approximation such as was considered in [27] for one-dimensional problems.

Another choice was made in this article. Namely, we chose to perform the circulant approximation in the $x$- and $y$-directions since, for the hexagonal plates considered here, these are the largest dimensions. However, for hexagonal prisms of larger aspect ratio, superior results may be gained by choosing the longitudinal ($z$) axis as one of the circulant approximation directions.

Finally, we remark upon the memory consumption of the preconditioner and the GMRES Krylov subspace. The memory consumption of the preconditioner scales as $O(x^4)$ which was not problematic for the simulations performed here. However, for higher resolution and/or larger size parameter simulations, it may be desirable to compress the preconditioner in some way. This was seen to be achievable for the 1-level preconditioner in [16] so it likely that some compression can also be obtained in the 2-level case, which may in turn lead to faster assembly times.

The iterative solves with GMRES were seen to be more reliable than those with BiCG-Stab, however much more memory intensive due to the storage of the entire Krylov subspace. The entire Krylov subspace was retained here since it guarantees convergence of solver, but it is not necessary. Experimentation with GMRES with (deflated) restarts (e.g., [28]) would be useful to determine a reliable, fast, and memory efficient strategy for performing the preconditioned solves with GMRES.

**Funding**

This work was supported by a grant from Skoltech as part of the Skoltech- MIT Next Generation Program, and the Design for Manufacturability (DFM) Methods, PDK Extensions, and Tools for Photonic Systems project sponsored by AIM Photonics.

**References**

[1] E. M. Purcell and C. R. Pennypacker, “Scattering and absorption of light by nonspherical dielectric grains,” *The Astrophysical Journal*, vol. 186, pp. 705–714, 1973.
[2] B. T. Draine and P. J. Flatau, “Discrete-dipole approximation for scattering calculations,” *JOSA A*, vol. 11, no. 4, pp. 1491–1499, 1994.

[3] G. H. Goedecke and S. G. O’Brien, “Scattering by irregular inhomogeneous particles via the digitized Green’s function algorithm,” *Applied Optics*, vol. 27, no. 12, pp. 2431–2438, 1988.

[4] M. A. Yurkin, V. P. Maltsev, and A. G. Hoekstra, “The discrete dipole approximation for simulation of light scattering by particles much larger than the wavelength,” *Journal of Quantitative Spectroscopy and Radiative Transfer*, vol. 106, no. 1-3, pp. 546–557, 2007.

[5] T. Nousiainen, E. Zubko, J. V. Niemi, K. Kupiainen, M. Lehtinen, K. Muinonen, and G. Videen, “Single-scattering modeling of thin, birefringent mineral-dust flakes using the discrete-dipole approximation,” *Journal of Geophysical Research: Atmospheres*, vol. 114, no. D7, 2009.

[6] A. Polimeridis, J. F. Villena, L. Daniel, and J. White, “Stable FFT-JVIE solvers for fast analysis of highly inhomogeneous dielectric objects,” *Journal of Computational Physics*, vol. 269, pp. 280–296, 2014.

[7] M. A. Yurkin, K. A. Semyanov, P. A. Tarasov, A. V. Chernyshev, A. G. Hoekstra, and V. P. Maltsev, “Experimental and theoretical study of light scattering by individual mature red blood cells by use of scanning flow cytometry and a discrete dipole approximation,” *Applied Optics*, vol. 44, no. 25, pp. 5249–5256, 2005.

[8] T. Nieminen, V. L. Loke, A. B. Stilgoe, G. Knöner, A. M. Brańczyk, N. R. Heckenberg, and H. Rubinsztein-Dunlop, “Optical tweezers computational toolbox,” *Journal of Optics A: Pure and Applied Optics*, vol. 9, no. 8, p. S196, 2007.

[9] P. Flatau and B. Draine, “Light scattering by hexagonal columns in the discrete dipole approximation,” *Optics Express*, vol. 22, no. 18, pp. 21834–21846, 2014.

[10] T. Sarkar, E. Arvas, and S. Rao, “Application of FFT and the conjugate gradient method for the solution of electromagnetic radiation from electrically large and small conducting bodies,” *IEEE Transactions on Antennas and Propagation*, vol. 34, no. 5, pp. 635–640, 1986.

[11] J. Goodman, B. T. Draine, and P. J. Flatau, “Application of fast-Fourier-transform techniques to the discrete-dipole approximation,” *Optics Letters*, vol. 16, no. 15, pp. 1198–1200, 1991.

[12] T. F. Chan and J. A. Olkin, “Circulant preconditioners for Toeplitz-block matrices,” *Numerical Algorithms*, vol. 6, no. 1, pp. 89–101, 1994.

[13] T. F. Chan, “An optimal circulant preconditioner for Toeplitz systems,” *SIAM Journal on Scientific and Statistical Computing*, vol. 9, no. 4, pp. 766–771, 1988.

[14] S. P. Groth, J. White, and A. G. Polimeridis, “Circulant preconditioning in the volume integral equation method for nanophotonics,” in *Applied Computational Electromagnetics Society Symposium (ACES), 2018 International*, pp. 1–2, IEEE, 2018.

[15] A. Tambova, S. P. Groth, J. K. White, and A. G. Polimeridis, “Adiabatic absorbers in photonics simulations with the volume integral equation method,” *IEEE Journal of Lightwave Technology*, vol. 36, no. 17, pp. 3765–3777, 2018.
[16] S. P. Groth, A. G. Polimeridis, A. Tambova, and J. K. White, “Circulant preconditioning in the volume integral equation method for silicon photonics,” in *eprint arXiv:1903.07884*, Mar. 2019.

[17] M. K. Ng, *Iterative methods for Toeplitz systems*. Numerical Mathematics and Scie, 2004.

[18] M. K. Ng and J. Pan, “Approximate Inverse Circulant-plus-Diagonal Preconditioners for Toeplitz-plus-Diagonal Matrices,” *SIAM Journal on Scientific Computing*, vol. 32, no. 3, pp. 1442–1464, 2010.

[19] P. J. Flatau, “Improvements in the discrete-dipole approximation method of computing scattering and absorption,” *Optics Letters*, vol. 22, no. 16, pp. 1205–1207, 1997.

[20] B. Carpentieri, “Fast preconditioned Krylov methods for boundary integral equations in electromagnetic scattering,” in *Trends in Electromagnetism-From Fundamentals to Applications*, InTech, 2012.

[21] P. Coulier, H. Pouransari, and E. Darve, “The Inverse Fast Multipole Method: Using a Fast Approximate Direct Solver as a Preconditioner for Dense Linear Systems,” *SIAM Journal on Scientific Computing*, vol. 39, no. 3, pp. A761–A796, 2017.

[22] P. C. Chaumet, A. Sentenac, and A. Rahmani, “Coupled dipole method for scatterers with large permittivity,” *Physical Review E*, vol. 70, no. 3, p. 036606, 2004.

[23] J. Markkanen and P. Ylä-Oijala, “Numerical comparison of spectral properties of volume-integral-equation formulations,” *Journal of Quantitative Spectroscopy and Radiative Transfer*, vol. 178, pp. 269–275, 2016.

[24] V. Strela and E. Tyrtyshnikov, “Which circulant preconditioner is better?,” *Mathematics of Computation of the American Mathematical Society*, vol. 65, no. 213, pp. 137–150, 1996.

[25] G. Strang, “A proposal for Toeplitz matrix calculations,” *Studies in Applied Mathematics*, vol. 74, no. 2, pp. 171–176, 1986.

[26] M. Frigo and S. G. Johnson, “The design and implementation of FFTW3,” *Proceedings of the IEEE*, vol. 93, no. 2, pp. 216–231, 2005.

[27] R. Remis, “Circulant preconditioners for domain integral equations in electromagnetics,” in *Electromagnetics in Advanced Applications (ICEAA), 2012 International Conference on*, pp. 337–340, IEEE, 2012.

[28] R. B. Morgan, “GMRES with deflated restarting,” *SIAM Journal on Scientific Computing*, vol. 24, no. 1, pp. 20–37, 2002.