Sweeping Preconditioner for the Helmholtz Equation: Hierarchical Matrix Representation

BJÖRN ENGQUIST
Department of Mathematics and ICES, University of Texas at Austin

LEXING YING
Department of Mathematics and ICES, University of Texas at Austin

Abstract

The paper introduces the sweeping preconditioner, which is highly efficient for iterative solutions of the variable-coefficient Helmholtz equation including very-high-frequency problems. The first central idea of this novel approach is to construct an approximate factorization of the discretized Helmholtz equation by sweeping the domain layer by layer, starting from an absorbing layer or boundary condition. Given this specific order of factorization, the second central idea is to represent the intermediate matrices in the hierarchical matrix framework. In two dimensions, both the construction and the application of the preconditioners are of linear complexity. The generalized minimal residual method (GMRES) solver with the resulting preconditioner converges in an amazingly small number of iterations, which is essentially independent of the number of unknowns. This approach is also extended to the three-dimensional case with some success. Numerical results are provided in both two and three dimensions to demonstrate the efficiency of this new approach. © 2011 Wiley Periodicals, Inc.

1 Introduction

This is the first of a series of papers on developing efficient preconditioners for the numerical solutions of the Helmholtz equation in two and three dimensions. The efficiency of preconditioners for the Helmholtz equation in the important high-frequency range are at present much lower than that of preconditioners for typical elliptic problems. This paper develops efficient preconditioners of the Helmholtz equation by exploiting the physical property of the wave phenomena and certain low-rank interaction properties of the Green's function.

Let the domain of interest be the unit box $D = (0,1)^d$ with $d = 2, 3$. The time-independent wave field $u(x)$ for $x \in D$ satisfies

$$\Delta u(x) + \frac{\omega^2}{c^2(x)} u(x) = f(x), \tag{1.1}$$

Communications on Pure and Applied Mathematics, Vol. LXIV, 0697–0735 (2011)

where ω is the angular frequency, $c(x)$ is the velocity field, and $f(x)$ is the external force. Commonly used boundary conditions are approximations of the Sommerfeld condition, which guarantees that the wave field generated by $f(x)$ propagates out of the domain to infinity. Other boundary conditions for part of the boundary will also be considered. By appropriately rescaling the system, it is convenient to assume that the mean of $c(x)$ is around 1. Then $\frac{\omega}{2\pi}$ is the typical wave number of this problem and $\lambda = \frac{2\pi}{\omega}$ is the typical wavelength.

The Helmholtz equation is ubiquitous since it is the root of almost all linear wave phenomena. Applications of the Helmholtz equation are abundant in acoustics, elasticity, electromagnetics, quantum mechanics, and geophysics. Therefore, efficient and accurate numerical solutions of the Helmholtz problem is one of the urgent problems in computational mathematics. This is, however, a very difficult problem for two main reasons. First, in a typical engineering application, the Helmholtz equation is discretized with at least 8 to 16 points per wavelength. Therefore, the number of samples n in each dimension is proportional to ω, the total number of samples N is $n^d = O(\omega^d)$, and the discrete system of the Helmholtz equation is of size $N \times N = O(\omega^d) \times O(\omega^d)$. In the high-frequency range when ω is large, this is an enormous system. Second, as the discrete system is highly indefinite and has a very oscillatory Green's function due to the wave nature of the Helmholtz equation, most of the modern multiscale techniques developed for elliptic or parabolic problems are no longer effective.

1.1 Approach and Contribution

In this paper, we propose a *sweeping preconditioner* for the iterative solution of the Helmholtz equation. In all examples, the Helmholtz equation is discretized by centered finite differences, i.e., the five-point stencil in 2D and the seven-point stencil in 3D.

In the 2D case, this new preconditioner is based on a block LDL^{T} factorization of the discrete Helmholtz operator. The overall process is to eliminate the unknowns layer by layer, starting from a layer where an approximation to the Sommerfeld condition is specified. The main observation is that each intermediate $n \times n$ Schur complement matrix of this block LDL^{T} factorization roughly corresponds to the restriction of a half-space Green's function to a line, and these Schur complement matrices are highly compressible with low-rank off-diagonal blocks. Representing and manipulating these matrices in the hierarchical matrix framework [7] requires only $O(n \log n)$ space and $O(n \log^2 n)$ steps. As a result, the block LDL^{T} factorization takes $O(n^2 \log^2 n) = O(N \log^2 N)$ steps. The resulting block LDL^{T} factorization serves as an excellent preconditioner for the discrete Helmholtz system, and applying it to any vector takes only $O(n^2 \log n) = O(N \log N)$ steps using again the hierarchical matrix framework. By combining this preconditioner with GMRES, we obtain iteration numbers that are almost independent of ω. In a typical example with a computational domain of

256 wavelengths in each dimension and four million unknowns, only three to four GMRES iterations are required (see Section 3).

We also extend this approach to the 3D case and construct an approximate block LDL^{T} factorization by eliminating the unknowns face by face, starting from a face at which an approximation to the Sommerfeld condition is specified. Though each intermediate $n^2 \times n^2$ Schur complement matrix corresponds to the restriction of a half-space Green's function to a face, the off-diagonal parts may not be as numerically low-rank as in 2D. However, since the goal is to construct a preconditioner, we still represent and manipulate these matrices under the hierarchical matrix framework. Numerical results show that applying the resulting preconditioner is highly efficient and the preconditioned GMRES solver converges in a small number of iterations, weakly depending on ω.

The main observation of the sweeping preconditioner comes from the analytic low-rank property of the Green's function of the *continuous* Helmholtz operator. On the other hand, the algorithms construct directly an approximation to the Green's function of the *discrete* Helmholtz operator. It is important that this Green's function be calculated from the discretized problem to be solved numerically and not be an independent approximation of the continuous analogue.

1.2 Related Work

There has been a vast literature on developing efficient numerical algorithms for the Helmholtz equation. A wide class of methods for special sets of solutions are based on asymptotic expansion of the solution $u(x)$. These techniques of geometric optics type are efficient when ω is very large. A review article on these methods can be found in [16]. There is also a class of methods based on boundary integral or volumetric integral representations. These integral equation methods can be highly efficient for piecewise constant velocity fields when combined with fast summation methods such as the fast multipole methods and the fast Fourier transforms [6, 9, 17, 18, 39, 40]. Here, however, we will focus on the methods that discretize the Helmholtz equation directly.

The most efficient direct methods for solving the discretized Helmholtz systems are the multifrontal methods or their pivoted versions [12, 25, 33]. The multifrontal methods exploit the locality of the discrete operator and construct an LDL^{T} factorization based on a hierarchical partitioning of the domain. Their computational costs depend quite strongly on the dimensions of the problem. In 2D, for a problem with $N = n \times n$ unknowns, a multifrontal method takes $O(N^{3/2})$ steps and $O(N \log N)$ storage space. The prefactor is usually rather small, making the multifrontal methods effectively the default choice for the 2D Helmholtz problem. In 3D, for a problem with $N = n \times n \times n$ unknowns, a multifrontal method takes $O(n^6) = O(N^2)$ steps and $O(n^4) = O(N^{4/3})$ storage space, making it very costly for large-scale 3D problems.

In the setting of the elliptic operators, the intermediate matrices of the multifrontal methods can be well approximated using hierarchical matrix algebra, and

this allows one to bring the cost down to linear complexity in both 2D and 3D [36, 46]. This is, however, not true for the Helmholtz operator. The sweeping preconditioner introduced in this paper is also based on constructing an LDL^* factorization of the Helmholtz operator. However, due to its specific sweeping (or elimination) order, which is very different from the one of the multifrontal methods, we are able to represent the intermediate matrices in a more effective way and obtain a highly efficient preconditioner.

There have been a number of developments on iterative methods for solving the Helmholtz equation. The following discussion is by no means complete and more details can be found in [20].

Standard multigrid methods do not work well for the Helmholtz equation for several reasons. The most important one is that the oscillations on the scale of the wavelength cannot be carried on the coarse grids. Several methods have been proposed to remedy this [8, 13, 23, 31, 34, 45]. For example, in [8, 34] Brandt and Livshits propose the wave-ray method. This method uses the standard smoothers to remove the coarse and fine components of the residue and then decomposes the component that oscillates on the scale of the wavelength into rays pointing at different directions. Each ray is further represented as the product of a smooth amplitude and an oscillatory phase, and the amplitude is removed by applying relaxation on an anisotropic grid aligned with the ray direction. A limitation of the wave-ray method, however, is that the method is essentially restricted only to the constant-velocity field. We would like to point out that there is a connection between the wave-ray method and the sweeping preconditioner proposed in this paper, as both methods exploit the analytic behavior of the Green's function of the Helmholtz equation: the wave-ray method relies on the Green's function over the whole domain, while the sweeping preconditioner uses its restriction to a single layer.

Several other methods [2, 11, 44] leverage the idea of domain decomposition. These methods are typically quite suitable for parallel implementation, as the computation in each subdomain can essentially be done independently. However, convergence rates of these methods are usually quite slow [20].

Another class of methods [1, 21, 22, 30] that has attracted a lot of attention recently seeks to precondition the Helmholtz operator with a shifted Laplacian operator,

$$\Delta - \frac{\omega^2}{c^2(x)}(\alpha + i\beta), \quad \alpha > 0,$$

to improve the spectral property of the discrete Helmholtz system. Since the shifted Laplacian operator is elliptic, standard algorithms such as multigrid can be used for the inversion of the above operator. These methods offer quite significant improvements for the convergence rate, but the reported number of iterations typically still grows linearly with respect to ω and is much larger than the iteration numbers produced by the sweeping preconditioner introduced in this paper.

Several other constructions of preconditioners [3, 24, 38] are based on incomplete LU (ILU) decomposition, i.e., generating only a small portion of the entries of the LU factorization of the discrete Helmholtz operator and applying this ILU decomposition as a preconditioner. Recent approaches based on ILUT (incomplete LU factorization with thresholding) and ARMS (algebraic recursive multilevel solver) have been reported in [38]. These ILU preconditioners bring down the number of iterations quite significantly; however, the number of iterations still scales linearly in ω. In connection with the ILU preconditioners, one can in fact view the sweeping preconditioner of this paper as an approximate LU (ALU) preconditioner: instead of keeping only a few selected entries, it approximates the whole inverse operator more accurately in a more sophisticated and effective form, thus resulting in substantially better convergence properties.

1.3 Contents

The rest of this paper is organized as follows. Section 2 presents the sweeping preconditioner in the 2D case, and Section 3 reports the 2D numerical results. We extend this approach to the 3D case in Section 4 and report the 3D numerical results in Section 5. Finally, Section 6 discusses some future directions of this work.

2 Preconditioner in 2D

2.1 Discretization

Recall that the computational domain is $D = (0,1)^2$. Let us assume for simplicity that the Sommerfeld condition is specified at all directions. One standard way of incorporating the Sommerfeld boundary condition into (1.1) is to use the perfectly matched layer (PML) [4, 10, 29]. Introduce

$$\sigma(t) = \begin{cases} \frac{C}{\eta} \cdot \left(\frac{t-\eta}{\eta}\right)^2, & t \in [0,\eta], \\ 0, & t \in [\eta, 1-\eta], \\ \frac{C}{\eta} \cdot \left(\frac{t-1+\eta}{\eta}\right)^2, & t \in [1-\eta, 1], \end{cases} \tag{2.1}$$

and

$$s_1(x_1) = \left(1 + i\frac{\sigma(x_1)}{\omega}\right)^{-1}, \quad s_2(x_2) = \left(1 + i\frac{\sigma(x_2)}{\omega}\right)^{-1}.$$

Here η is typically about one wavelength and C is an appropriate positive constant independent of ω. The PML approach replaces ∂_1 with $s_1(x_1)\partial_1$ and ∂_2 with $s_2(x_2)\partial_2$, which effectively provides a damping layer of width η near the boundary of the domain $[0,1]^2$. The resulting equation is

$$\left((s_1\partial_1)(s_1\partial_1) + (s_2\partial_2)(s_2\partial_2) + \frac{\omega^2}{c^2(x)}\right)u = f, \quad x \in D = [0,1]^2,$$
$$u = 0, \quad x \in \partial D.$$

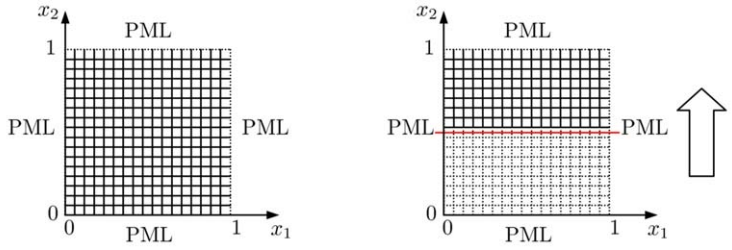


FIGURE 2.1. Left: Discretization grid in 2D. Right: Sweeping order in 2D. The dotted grid indicates the unknowns that have already been eliminated.

Without loss of generality, we assume that $f(x)$ is supported inside $[\eta, 1-\eta]^2$ (away from the PML). Dividing the above equation by $s_1(x_1)s_2(x_2)$ results in

$$\left(\partial_1\left(\frac{s_1}{s_2}\partial_1\right) + \partial_2\left(\frac{s_2}{s_1}\partial_2\right) + \frac{\omega^2}{s_1 s_2 \cdot c^2(x)}\right) u = f.$$

The advantage of working with this equation is that it is symmetric, which offers some convenience from the algorithmic point of view. We discretize the domain with a Cartesian grid with spacing $h = 1/(n+1)$, where n is assumed be an integer power of 2 for simplicity. In order to discretize each wavelength with a couple of points, the number n of points in each dimension needs to be proportional to ω. The interior points of this grid are

$$P = \{p_{i,j} = (ih, jh) : 1 \le i, j \le n\}$$

(see Figure 2.1 (left)), and the total number of points N is equal to n^2.

We denote by $u_{i,j}$, $f_{i,j}$, and $c_{i,j}$ the values of $u(x)$, $f(x)$, and $c(x)$ at point $p_{i,j} = (ih, jh)$. The standard five-point stencil finite difference method writes down the equation at points in P using central difference. The resulting equation at $p_{i,j} = (ih, jh)$ is

$$\begin{aligned} &\frac{1}{h^2}\left(\frac{s_1}{s_2}\right)_{i-\frac{1}{2},j} u_{i-1,j} + \frac{1}{h^2}\left(\frac{s_1}{s_2}\right)_{i+\frac{1}{2},j} u_{i+1,j} \\ &\quad + \frac{1}{h^2}\left(\frac{s_2}{s_1}\right)_{i,j-\frac{1}{2}} u_{i,j-1} + \frac{1}{h^2}\left(\frac{s_2}{s_1}\right)_{i,j+\frac{1}{2}} u_{i,j+1} \\ &\quad + \left(\frac{\omega^2}{(s_1 s_2)_{i,j} \cdot c_{i,j}^2} - (\cdots)\right) u_{i,j} = f_{i,j} \end{aligned} \tag{2.2}$$

with $u_{i',j'}$ equal to 0 for (i', j') that violates $1 \le i', j' \le n$. Here $(\cdots)$ stands for the sum of the four coefficients appearing in the first four terms. We order $u_{i,j}$ row by row starting from the first row $j = 1$ and denote the vector containing all unknowns by

$$u = (u_{1,1}, u_{2,1}, \ldots, u_{n,1}, \ldots, u_{1,n}, u_{2,n}, \ldots, u_{n,n})^{\mathsf{T}}.$$

Similarly, $f_{i,j}$ are ordered in the same way and the vector f is

$$f = (f_{1,1}, f_{2,1}, \ldots, f_{n,1}, \ldots, f_{1,n}, f_{2,n}, \ldots, f_{n,n})^{\mathsf{T}}.$$

By denoting the linear operator in (2.2) by A, we obtain the linear system $Au = f$. We further define P_m to be the unknowns in the m^{th} row

$$P_m = \{p_{1,m}, \ldots, p_{n,m}\}$$

and introduce

$$u_m = (u_{1,m}, u_{2,m}, \ldots, u_{n,m})^{\mathsf{T}} \quad \text{and} \quad f_m = (f_{1,m}, f_{2,m}, \ldots, f_{n,m})^{\mathsf{T}}.$$

Then

$$u = (u_1^{\mathsf{T}}, u_2^{\mathsf{T}}, \ldots, u_n^{\mathsf{T}})^{\mathsf{T}} \quad \text{and} \quad f = (f_1^{\mathsf{T}}, f_2^{\mathsf{T}}, \ldots, f_n^{\mathsf{T}})^{\mathsf{T}}.$$

Using this notation, the system $Au = f$ takes the following tridiagonal block form

$$\begin{pmatrix} A_{1,1} & A_{1,2} & & & \\ A_{2,1} & A_{2,2} & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & A_{n-1,n} \\ & & & A_{n,n-1} & A_{n,n} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ \vdots \\ u_n \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ \vdots \\ f_n \end{pmatrix},$$

where $A_{m,m}$ are tridiagonal matrices and $A_{m,m-1} = A_{m-1,m}^{\mathsf{T}}$ are diagonal matrices.

We introduce the notion of *sweeping factorization*, which is essentially a block LDL^{T} factorization of A that eliminates the unknowns layer by layer. Starting the elimination from the first row P_1 gives

$$A = L_1 \begin{pmatrix} S_1 & & & & \\ & S_2 & A_{2,3} & & \\ & A_{3,2} & \ddots & \ddots & \\ & & \ddots & \ddots & \ddots \end{pmatrix} L_1^{\mathsf{T}},$$

where $S_1 = A_{1,1}$, $S_2 = A_{2,2} - A_{2,1}S_1^{-1}A_{1,2}$, and the matrix L_1 is a block lower-triangular matrix given by

$$L_1(P_2, P_1) = A_{2,1}S_1^{-1}, \quad L_1(P_i, P_i) = I, \ \ 1 \le i \le n, \quad \text{and zero otherwise.}$$

Repeating this process over all P_m for $m = 2, \ldots, n-1$ gives

$$A = L_1 \cdots L_{n-1} \begin{pmatrix} S_1 & & & \\ & S_2 & & \\ & & \ddots & \\ & & & S_n \end{pmatrix} L_{n-1}^{\mathsf{T}} \cdots L_1^{\mathsf{T}}, \tag{2.3}$$

where $S_m = A_{m,m} - A_{m,m-1} S_{m-1}^{-1} A_{m-1,m}^{\mathsf{T}}$ for $m = 2, 3, \ldots, n$. The matrix L_m is given by

$$L_m(P_{m+1}, P_m) = A_{m+1,m} S_m^{-1}, \quad L_m(P_i, P_i) = I, \ \ 1 \le i \le n,$$
$$\text{and zero otherwise.}$$

This process is illustrated graphically in Figure 2.1 (right). Inverting this factorization (2.3) for A gives the following formula for u:

$$u = (L_1^{\mathsf{T}})^{-1} \cdots (L_{n-1}^{\mathsf{T}})^{-1} \begin{pmatrix} S_1^{-1} & & & \\ & S_2^{-1} & & \\ & & \ddots & \\ & & & S_n^{-1} \end{pmatrix} L_{n-1}^{-1} \cdots L_1^{-1} f.$$

Algorithmically, the construction of the sweeping factorization of A can be summarized as follows by introducing $T_m = S_m^{-1}$.

Algorithm 2.1. Construction of the sweeping factorization of A.

1: $S_1 = A_{1,1}$ and $T_1 = S_1^{-1}$.
2: **for** $m = 2, \ldots, n$ **do**
3: $\quad S_m = A_{m,m} - A_{m,m-1} T_{m-1} A_{m-1,m}$ and $T_m = S_m^{-1}$.
4: **end for**

Since S_m and T_m are in general dense matrices of size $n \times n$, the cost of the construction algorithm is of order $O(n^4) = O(N^2)$. The computation of $u = A^{-1} f$ is carried out in the following algorithm once the sweeping factorization is ready.

Algorithm 2.2. Computation of $u = A^{-1} f$ using the sweeping factorization of A.

1: **for** $m = 1, \ldots, n$ **do**
2: $\quad u_m = f_m$
3: **end for**
4: **for** $m = 1, \ldots, n-1$ **do**
5: $\quad u_{m+1} = u_{m+1} - A_{m+1,m}(T_m u_m)$
6: **end for**
7: **for** $m = 1, \ldots, n$ **do**
8: $\quad u_m = T_m u_m$
9: **end for**
10: **for** $m = n-1, \ldots, 1$ **do**
11: $\quad u_m = u_m - T_m(A_{m,m+1} u_{m+1})$
12: **end for**

Obviously the computations of $T_m u_m$ in the second and the third loops only need to be carried out once. However, we choose to write the algorithm this way for simplicity. The cost of computing u is of order $O(n^3) = O(N^{3/2})$, and this is

$O(N^{1/2})$ times more expensive compared to the multifrontal methods. Therefore Algorithms 2.1 and 2.2 are not very useful in this simple form.

2.2 Main Observation

Let us consider the meaning of the matrix $T_m = S_m^{-1}$. Consider only the top-left $m \times m$ blocks of the factorization (2.3).

$$\begin{pmatrix} A_{1,1} & A_{1,2} & & & \\ A_{2,1} & A_{2,2} & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & A_{m-1,m} \\ & & & A_{m-1,m} & A_{m,m} \end{pmatrix} = L_1 \cdots L_{m-1} \begin{pmatrix} S_1 & & & \\ & S_2 & & \\ & & \ddots & \\ & & & S_m \end{pmatrix} L_{m-1}^{\mathsf{T}} \cdots L_1^{\mathsf{T}}, \tag{2.4}$$

where the L_k-matrices are taken to be their restrictions to the top-left $m \times m$ blocks. The matrix on the left is in fact the discrete Helmholtz operator of the half-space problem below $x_2 = (m+1)h$ and with the zero boundary condition on $x_2 = (m+1)h$. Inverting the above factorization gives

$$\begin{pmatrix} A_{1,1} & A_{1,2} & & & \\ A_{2,1} & A_{2,2} & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & A_{m-1,m} \\ & & & A_{m,m-1} & A_{m,m} \end{pmatrix}^{-1} = (L_1^{\mathsf{T}})^{-1} \cdots (L_{m-1}^{\mathsf{T}})^{-1} \begin{pmatrix} S_1^{-1} & & & \\ & S_2^{-1} & & \\ & & \ddots & \\ & & & S_m^{-1} \end{pmatrix} L_{m-1}^{-1} \cdots L_1^{-1}. \tag{2.5}$$

The matrix on the left is the discrete half-space Green's function of the Helmholtz operator with zero boundary condition. On the right side, due to the definition of the matrices $L_1, \ldots, L_{m-1}$, the $(m,m)^{\text{th}}$ block of the whole product is exactly equal to S_m^{-1}. Therefore, *$T_m = S_m^{-1}$ is the restriction to $x_2 = mh$ of the discrete half-space Green function of the Helmholtz operator with zero boundary at $x_2 = (m+1)h$.*

The main observation of our approach is that *T_m and S_m are highly compressible with numerically low-rank off-diagonal blocks.* The following theorem shows that

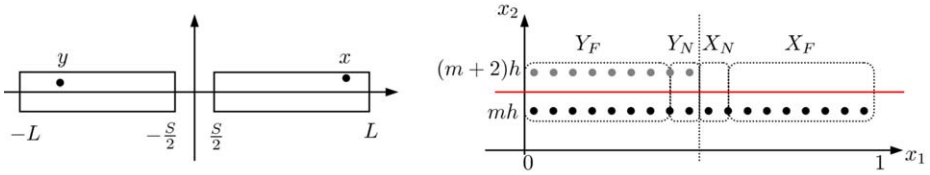


FIGURE 2.2. Left: The setting of Theorem 2.4. Right: The setting of Theorem 2.3.

this is true for the continuous half-space Green's function for the case of constant velocity field $c(x) = 1$.

THEOREM 2.3 *Let*

$$Y = \{p_{i,m} = (ih, mh), i = 1, \ldots, \tfrac{n}{2}\},$$
$$X = \{p_{i,m} = (ih, mh), i = \tfrac{n}{2} + 1, \ldots, n\},$$

and G be the (continuous) half-space Green's function of the Helmholtz operator for the domain $(-\infty, \infty) \times (-\infty, (m+1)h)$ with zero boundary condition. Then $(G(x, y))_{x \in X, y \in Y}$ is numerically low-rank. More precisely, for any $\varepsilon > 0$, there exist a constant $R = O(\log \omega |\log \varepsilon|^2)$ and functions $\{\alpha_r(x)\}_{1 \le r \le R}$ for $x \in X$ and functions $\{\beta_r(y)\}_{1 \le r \le R}$ for $y \in Y$ such that

$$\left| G(x, y) - \sum_{r=1}^{R} \alpha_r(x) \beta_r(y) \right| \le \varepsilon \quad \text{for } x \in X,\ y \in Y.$$

The proof of this theorem relies on the following theorem from [37]. Let $H_0(\cdot)$ be the zeroth-order Hankel function of the first kind.

THEOREM 2.4 *Let ω be the angular frequency and $\lambda = 2\pi/\omega$. Let $W > 0$. There exists $C(W)$ such that, for $L > 0$, $\varepsilon > 0$, and $S > C(W)|\log \varepsilon|\lambda$, there exist a constant $J \le \log(\omega L)|\log \varepsilon|^2$, functions $\{\phi_j(x)\}_{1 \le j \le J}$, and functions $\{\chi_j(y)\}_{1 \le j \le J}$ such that*

$$\left| H_0(\omega |x - y|) - \sum_{j=1}^{J} \phi_j(x) \chi_j(y) \right| \le \varepsilon$$

for

$$y \in \left[-L, -\tfrac{S}{2}\right] \times \left[-\tfrac{W}{2}, \tfrac{W}{2}\right] \quad \text{and} \quad x \in \left[\tfrac{S}{2}, L\right] \times \left[-\tfrac{W}{2}, \tfrac{W}{2}\right].$$

The setting of this theorem is illustrated in Figure 2.2 (left). Using Theorem 2.4, the proof of Theorem 2.3 goes as follows.

PROOF OF THEOREM 2.3. Let $W = 2h$. We partition the set X into the union of the near set X_N and the far set X_F depending on the distance from Y:

$$X_N = \{p = (p_1, p_2) \in X,\ p_1 \le \tfrac{1}{2} + \tfrac{1}{2} C(W) |\log(\tfrac{\varepsilon}{2})| \lambda\},$$
$$X_F = \{p = (p_1, p_2) \in X,\ p_1 > \tfrac{1}{2} + \tfrac{1}{2} C(W) |\log(\tfrac{\varepsilon}{2})| \lambda\}.$$

Similarly, Y is partitioned into the union of Y_N and Y_F:

$$Y_N = \{p = (p_1, p_2) \in Y,\ p_1 \geq \tfrac{1}{2} - \tfrac{1}{2}C(W)|\log(\tfrac{\varepsilon}{2})|\lambda\},$$
$$Y_F = \{p = (p_1, p_2) \in Y,\ p_1 < \tfrac{1}{2} - \tfrac{1}{2}C(W)|\log(\tfrac{\varepsilon}{2})|\lambda\}.$$

See Figure 2.2 (right). These partitionings introduce a natural block structure for the matrix $(G(x, y))_{x\in X, y\in Y}$:

$$\begin{pmatrix} (G(x,y))_{x\in X_N, y\in Y_N} & (G(x,y))_{x\in X_N, y\in Y_F} \\ (G(x,y))_{x\in X_F, y\in Y_N} & (G(x,y))_{x\in X_F, y\in Y_F} \end{pmatrix}. \tag{2.6}$$

Let $q = \frac{\lambda}{h}$ be the number of points per wavelength. It is clear from the definition of X_N and Y_N that each of them has at most $\frac{1}{2}C(W)|\log\varepsilon|\frac{\lambda}{h} = \frac{1}{2}C(2h)|\log\varepsilon|q$ points. Hence the ranks of the $(1,1)$, $(1,2)$, and $(2,1)$ blocks of (2.6) are all bounded from above by $\frac{1}{2}C(2h)|\log\varepsilon|q$.

Let us consider the $(2,2)$ block. Define $M(Y_F)$ to be the mirror image set of the set Y_F with respect to the line $x_2 = (m+1)h$. Due to the zero Dirichlet boundary condition at $x_2 = (m+1)h$, for $x \in X_F$ and $y \in Y_F$

$$G(x, y) = H_0(\omega|x - y|) - H_0(\omega|x - M(y)|)$$

where $M(y) \in M(Y_F)$ is the mirror image of y. $Y_F \cup M(Y_F)$ is contained in the box

$$\left[0, \tfrac{1}{2} - \tfrac{1}{2}C(W)|\log(\tfrac{\varepsilon}{2})|\lambda\right] \times [mh, (m+2)h]$$

and X_F is in

$$\left[\tfrac{1}{2} + \tfrac{1}{2}C(W)|\log(\tfrac{\varepsilon}{2})|\lambda, 1\right] \times [mh, (m+2)h].$$

Since the distance between these two boxes is $C(W)|\log(\frac{\varepsilon}{2})|\lambda$ and their widths are bounded by 1, applying Theorem 2.4 shows that there exist a constant $J \leq \log(\omega)|\log(\frac{\varepsilon}{2})|^2$, functions $\{\phi_j(x)\}_{1\leq j\leq J}$, and functions $\{\chi_j(y)\}_{1\leq j\leq J}$ such that

$$\left|H_0(\omega|x - y|) - \sum_{j=1}^{J} \phi_j(x)\chi_j(y)\right| \leq \frac{\varepsilon}{2}$$

for $x \in X_F$ and $y \in Y_F \cup M(Y_F)$. This implies that

$$\left|G(x, y) - \sum_{j=1}^{J} \phi_j(x)(\chi_j(y) - \chi_j(M(y)))\right| \leq \varepsilon.$$

Combining this with the estimates for the other three blocks shows that there exist $R = \frac{3}{2}C(2h)|\log(\frac{\varepsilon}{2})|q + \log(\omega)|\log(\frac{\varepsilon}{2})|^2 = O(\log(\omega)|\log\varepsilon|^2)$ and functions $\{\alpha_r(x)\}_{1\leq r\leq R}$ for $x \in X$ and functions $\{\beta_r(y)\}_{1\leq r\leq R}$ for $y \in Y$ such that

$$\left|G(x, y) - \sum_{r=1}^{R} \alpha_r(x)\beta_r(y)\right| \leq \varepsilon \quad \text{for } x \in X,\ y \in Y.$$

□

For a fixed ε, Theorem 2.3 shows that the rank R grows logarithmically with respect to ω (and thus n). Though the theorem states the result under the case that X contains the points on the left half and Y contains the points on the right half, it also applies to any disjoint intervals X and Y on $x_2 = mh$ due to the translational invariance of the kernel $G(x, y)$ in the x_1-direction. It is also clear that when X and Y are well separated from each other, the actual rank R should be smaller.

Theorem 2.3 can be extended to the case of smooth layered media where the velocity variation only depends on x_1. In this case, the restriction of the Green's function to $x_2 = mh$ does not develop caustics. Therefore, the geometric optics representation $A(x, y)e^{i\omega\Phi(x,y)}$ of the Green's function for $x \in X$ and $y \in Y$ can be made sufficiently accurate as long as X and Y are well separated. The amplitude $A(x, y)$ is numerically low-rank due to its smoothness. The phase term is also numerically low-rank since for the layered media $\Phi(x, y) = \tau(x) - \tau(y)$ where $\tau(\cdot)$ is the travel time function from a fixed point. Therefore, as their product, the Green's function $G(x, y)$ is also numerically low-rank for well-separated X and Y.

Numerical experiments confirm the statement of Theorem 2.3. For the constant-coefficient case $c(x) = 1$ with $\frac{\omega}{2\pi} = 32$ ($n = 256$), Figure 2.3 (left) shows the numerical ranks of the off-diagonal blocks of T_m for $m = 128$. For each off-diagonal block, the singular values of the block are calculated and the color of the block in Figure 2.3 indicates the number of singular values that are greater than 10^{-6}. For nonconstant velocity fields $c(x)$, the rank estimate would depend on the variations in $c(x)$, and numerical results suggest that the off-diagonal blocks of T_m and S_m still admit this low-rankness property for a wide class of $c(x)$. An example for the nonconstant velocity field is given in Figure 2.3 (middle).

We would like to emphasize that both the Sommerfeld boundary condition and the layer-by-layer sweeping order are essential for this low-rank property. To illustrate that, we perform a test with the same threshold 10^{-6} but with zero Dirichlet boundary condition. The result of T_m for $m = 128$ is plotted in Figure 2.3 (right). It is clear that the ranks of the off-diagonal blocks are much higher and grow almost linearly with respect to the size of the block. This clearly shows the importance of the Sommerfeld boundary condition. A similar matrix T_m would also appear if one adopts different elimination orders such as the one of multifrontal methods or the one proposed in [36]; therefore these other elimination orders do not result in efficient solution methods for the Helmholtz equation.

2.3 Hierarchical Matrix Representation

Since the matrices T_m and S_m are highly compressible with numerically low-rank off-diagonal blocks, it is natural to represent these matrices using the hierarchical matrix (or H-matrix) framework proposed by Hackbusch et al. [7, 26, 27], where off-diagonal blocks are represented in low-rank factorized form. The discussion below is by no means original and is included for the sake of completeness.

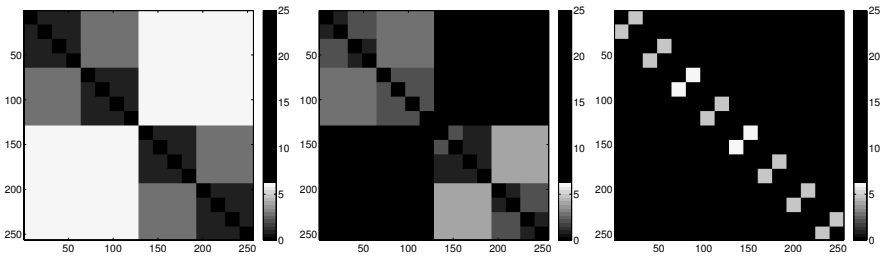


FIGURE 2.3. Numerical ranks of off-diagonal blocks of T_m. Left: Constant-coefficient case with PML boundary condition. Middle: Nonconstant-coefficient case with PML boundary condition. Right: Constant-coefficient case with zero Dirichlet boundary condition.

For each layer m, we construct a hierarchical decomposition of the grid points in P_m through bisection. At level 0 (the top level), the set

$$J_1^0 = P_m.$$

At level ℓ, there are 2^ℓ sets J_i^ℓ for $i = 1, 2, \ldots, 2^\ell$ given by

$$J_i^\ell = \{p_{t,m} : (i-1) \cdot n/2^\ell + 1 \le t \le i \cdot n/2^\ell\}.$$

The bisection is stopped when each set J_i^ℓ contains only a small number of indices. It is clear that the number of total levels L is equal to $\log_2 n - O(1)$ (see Figure 2.4 (left)). We often write $G(J_i^\ell, J_{i'}^\ell)$ (the restriction of a matrix G to J_i^ℓ and $J_{i'}^\ell$) as $G_{i,i'}^\ell$. The hierarchical matrix representation relies on the notion of *well separatedness* between different sets. If J_i^ℓ and $J_{i'}^\ell$ are well separated from each other, then $G(J_i^\ell, J_{i'}^\ell)$ is allowed to be stored in a low-rank factorized form.

There are two different choices of the notion of well separatedness [7]. In the *weakly admissible case* J_i^ℓ and $J_{i'}^\ell$ are well separated if and only if they are disjoint, while in the *strongly admissible case* J_i^ℓ and $J_{i'}^\ell$ are well separated if and only if the distance between them is greater than or equal to their width. Next, define the interaction list of J_i^ℓ to be the set of all index sets $J_{i'}^\ell$ such that J_i^ℓ is well separated from $J_{i'}^\ell$ but J_i^ℓ's parent is not well separated from $J_{i'}^\ell$'s parent. It is clear from this definition that being a member of another set's interaction list is a symmetric relationship.

Weakly Admissible Case

In the weakly admissible case, the interaction list of J_{2i}^ℓ contains only J_{2i-1}^ℓ and vice versa.

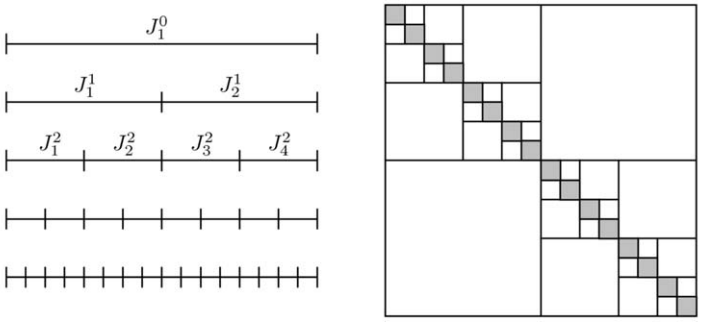


FIGURE 2.4. Hierarchical matrix representation. Left: Hierarchical partitioning of the index set P_m for each layer. Right: Induced partitioning of the matrix T_m in the weakly admissible case. Off-diagonal blocks (in white) are stored in low-rank factorized form. Diagonal blocks (in gray) are stored densely.

Matrix representation. First fix an error threshold ε. Let $R = O(\log \omega) = O(\log n)$ be the maximum over the ranks of the off-diagonal blocks on all levels. For a given matrix G, the hierarchical matrix framework represents all blocks $G_{i,i'}^{\ell} = G(J_i^{\ell}, J_{i'}^{\ell})$ with J_i^{ℓ} and $J_{i'}^{\ell}$ in each other's interaction list in the factorized form with rank less than or equal to R. For example, at the first level, the two off-diagonal blocks $G_{1,2}^1 = G(J_1^1, J_2^1)$ and $G_{2,1}^1 = G(J_2^1, J_1^1)$ are represented by

$$G_{1,2}^1 \approx U_{1,2}^1 (V_{1,2}^1)^{\mathsf{T}} \quad \text{and} \quad G_{2,1}^1 \approx U_{2,1}^1 (V_{2,1}^1)^{\mathsf{T}},$$

where each of $U_{1,2}^1$, $U_{2,1}^1$, $V_{1,2}^1$, and $V_{2,1}^1$ has at most R columns. At the second level, the new off-diagonal blocks are $G_{1,2}^2$, $G_{2,1}^2$, $G_{3,4}^2$, and $G_{4,3}^2$, each represented in a similar way. Finally, at level $L-1$, all diagonal blocks $G_{i,i}^{L-1}$ for $i = 1, \ldots, 2^{L-1}$ are stored densely. This representation is illustrated in Figure 2.4 (right). The total storage cost is $O(Rn \log n)$.

Matrix-vector multiplication. Let us consider the product Gf where f is a vector of size n. Denote by f_i^{ℓ} the part of f restricted to J_i^{ℓ}. Using the block matrix form, the product is

$$\begin{pmatrix} G_{1,1}^1 & G_{1,2}^1 \\ G_{2,1}^1 & G_{2,2}^1 \end{pmatrix} \begin{pmatrix} f_1^1 \\ f_2^1 \end{pmatrix} = \begin{pmatrix} G_{1,1}^1 f_1^1 + G_{1,2}^1 f_2^1 \\ G_{2,1}^1 f_1^1 + G_{2,2}^1 f_2^1 \end{pmatrix}.$$

First, the product $G_{1,2}^1 f_2^1$ is computed with $G_{1,2}^1 f_2^1 \approx U_{1,2}^1 ((V_{1,2}^1)^{\mathsf{T}} f_2^1)$. The same is carried out for the product $G_{2,1}^1 f_1^1$. Second, the computation of $G_{1,1}^1 f_1^1$ and $G_{2,2}^1 f_2^1$ is done recursively since both $G_{1,1}^1$ and $G_{2,2}^1$ are in the hierarchical matrix form. We denote by $\mathsf{hmatvec}(G, f)$ this matrix-vector multiplication procedure, and its computational cost is equal to $O(Rn \log n)$.

Matrix addition and subtraction. Consider the sum of two matrices G and H with their off-diagonal blocks given in the factorized form by $G^\ell_{i,j} \approx U^\ell_{i,j}(V^\ell_{i,j})^\mathsf{T}$ and $H^\ell_{i,j} \approx X^\ell_{i,j}(Y^\ell_{i,j})^\mathsf{T}$. Under the block matrix notation, the sum is

$$\begin{pmatrix} G^1_{1,1} & G^1_{1,2} \\ G^1_{2,1} & G^1_{2,2} \end{pmatrix} + \begin{pmatrix} H^1_{1,1} & H^1_{1,2} \\ H^1_{2,1} & H^1_{2,2} \end{pmatrix} = \begin{pmatrix} G^1_{1,1} + H^1_{1,1} & G^1_{1,2} + H^1_{1,2} \\ G^1_{2,1} + H^1_{2,1} & G^1_{2,2} + H^1_{2,2} \end{pmatrix}.$$

First, consider the off-diagonal block

$$G^1_{1,2} + H^1_{1,2} \approx U^1_{1,1}(V^1_{1,2})^\mathsf{T} + X^1_{1,2}(Y^1_{1,2})^\mathsf{T} = \big(U^1_{1,2}, X^1_{1,2}\big)\big(V^1_{1,2}, Y^1_{1,2}\big)^\mathsf{T}.$$

One needs to recompress the last two matrices in order to prevent the rank of the low-rank factorization from increasing indefinitely. This can be done by computing the QR decomposition of $(U^1_{1,2}, X^1_{1,2})$ and $(V^1_{1,2}, Y^1_{1,2})$, followed by a truncated SVD of a matrix of small size. The same procedure is carried out for $G^1_{2,1} + H^1_{2,1}$ to compute the necessary factorization. Second, consider the diagonal blocks. $G^1_{1,1} + H^1_{1,1}$ and $G^1_{2,2} + H^1_{2,2}$ are done recursively since they are two sums of the same nature but only half the size. This additional procedure is denoted by $\mathsf{hadd}(G, H)$. The subtraction procedure is almost the same and is denoted by $\mathsf{hsub}(G, H)$. Both of them take $O(R^2 n \log n)$ steps.

Matrix multiplication. Let us consider the sum of two matrices G and H with their off-diagonal blocks represented by $G^\ell_{i,j} \approx U^\ell_{i,j}(V^\ell_{i,j})^\mathsf{T}$ and $H^\ell_{i,j} \approx X^\ell_{i,j}(Y^\ell_{i,j})^\mathsf{T}$. Under the block matrix form, the product is

$$\begin{pmatrix} G^1_{1,1} & G^1_{1,2} \\ G^1_{2,1} & G^1_{2,2} \end{pmatrix} \cdot \begin{pmatrix} H^1_{1,1} & H^1_{1,2} \\ H^1_{2,1} & H^1_{2,2} \end{pmatrix} = \begin{pmatrix} G^1_{1,1}H^1_{1,1} + G^1_{1,2}H^1_{2,1} & G^1_{1,1}H^1_{1,2} + G^1_{1,2}H^1_{2,2} \\ G^1_{2,1}H^1_{1,1} + G^1_{2,2}H^1_{2,1} & G^1_{2,1}H^1_{1,2} + G^1_{2,2}H^1_{2,2} \end{pmatrix}.$$

First, the off-diagonal block

$$G^1_{1,1}H^1_{1,2} + G^1_{1,2}H^1_{2,2} \approx G^1_{1,1}X^1_{1,2}(Y^1_{1,2})^\mathsf{T} + U^1_{1,2}(V^1_{1,2})^\mathsf{T}H^1_{2,2}.$$

The computation $G^1_{1,1}X^1_{1,2}$ and $(V^1_{1,2})^\mathsf{T}H^1_{2,2}$ are essentially matrix-vector multiplications. Once they are done, the remaining computation is then similar to the off-diagonal part of the matrix addition algorithm. The other off-diagonal block $G^1_{2,1}H^1_{1,1} + G^1_{2,2}H^1_{2,1}$ is done in the same way.

Next, consider the diagonal blocks. Take $G^1_{1,1}H^1_{1,1} + G^1_{1,2}H^1_{2,1}$ as an example. The first part $G^1_{1,1}H^1_{1,1}$ is done using recursion. The second part is $G^1_{1,2}H^1_{2,1} \approx U^1_{1,2}(V^1_{1,2})^\mathsf{T}X^1_{2,1}(Y^1_{2,1})^\mathsf{T}$, where the middle product is carried out first in order to minimize the computational cost. The final sum $G^1_{1,1}H^1_{1,1} + G^1_{1,2}H^1_{2,1}$ is done using the matrix addition algorithm described above. The same procedure can be

carried out for $G^1_{2,1}H^1_{1,2} + G^1_{2,2}H^1_{2,2}$. This matrix multiplication procedure is denoted by $\mathsf{hmul}(G, H)$ and its computational cost is $O(R^2 n \log^2 n)$.

Matrix inversion. The inverse of G is done by performing a 2×2 block matrix inversion:

$$\begin{pmatrix} G^1_{1,1} & G^1_{1,2} \\ G^1_{2,1} & G^1_{2,2} \end{pmatrix}^{-1} = \begin{pmatrix} (G^1_{1,1})^{-1} + (G^1_{1,1})^{-1}G^1_{1,2}S^{-1}G^1_{2,1}(G^1_{1,1})^{-1} & -(G^1_{1,1})^{-1}G^1_{1,2}S^{-1} \\ -S^{-1}G^1_{2,1}(G^1_{1,1})^{-1} & S^{-1} \end{pmatrix}$$

where $S = G^1_{2,2} - G^1_{2,1}(G^1_{1,1})^{-1}G^1_{1,2}$. The computation of this formula requires matrix addition and multiplication, along with the inversion of two matrices S and $G^1_{1,1}$, half of the original size. The matrix addition and multiplication is carried out by the above procedures, while the inversions are done recursively. This matrix inversion procedure is denoted by $\mathsf{hinv}(G)$ and its cost is $O(R^2 n \log^2 n)$.

Multiplication with a diagonal matrix. Finally, we consider the multiplication of G by a diagonal matrix D. Denote the two diagonal blocks of D on the first level by $D^1_{1,1}$ and $D^1_{2,2}$, both of which are diagonal matrices. In the block matrix form, the product becomes

$$\begin{pmatrix} G^1_{1,1} & G^1_{1,2} \\ G^1_{2,1} & G^1_{2,2} \end{pmatrix} \cdot \begin{pmatrix} D^1_{1,1} & \\ & D^1_{2,2} \end{pmatrix} = \begin{pmatrix} G^1_{1,1}D^1_{1,1} & G^1_{1,2}D^1_{2,2} \\ G^1_{2,1}D^1_{1,1} & G^1_{2,2}D^1_{2,2} \end{pmatrix}.$$

Consider the off-diagonal blocks first. For example,

$$G^1_{1,2}D^1_{2,2} \approx U^1_{1,2}(V^1_{1,2})^{\mathsf{T}}D^1_{2,2},$$

and this is done by scaling each column of $(V^1_{1,2})^{\mathsf{T}}$ by the corresponding diagonal entries of $D^1_{2,2}$. The same is true for $G^1_{2,1}D^1_{1,1}$. For the diagonal blocks, say $G^1_{1,1}D^1_{1,1}$, we simply apply recursion since $G^1_{1,1}$ is itself a hierarchical matrix and $D^1_{1,1}$ is itself diagonal. This special multiplication procedure is denoted by $\mathsf{hdiagmul}(G, D)$ if D is on the right or $\mathsf{hdiagmul}(D, G)$ if D is on the left. The cost of both procedures is $O(Rn \log n)$.

Strongly Admissible Case

The matrix representation and operations in the strongly admissible case are similar to the ones in the weakly admissible case. The only one that requires significant modification is the matrix multiplication procedure $R = \mathsf{hmul}(G, H)$, where the most common step is the calculation of

$$R^{\ell}_{i,i''} \leftarrow G^{\ell}_{i,i'}H^{\ell}_{i',i''}. \tag{2.7}$$

In order to simplify the discussion, we denote a matrix symbolically by H if it is in hierarchical form and by F if it is represented in a factorized form. The product

(2.7) can then take one of the following eight forms:

$$\mathsf{H} = \mathsf{H}\cdot\mathsf{H}, \quad \mathsf{H} = \mathsf{H}\cdot\mathsf{F}, \quad \mathsf{H} = \mathsf{F}\cdot\mathsf{H}, \quad \mathsf{H} = \mathsf{F}\cdot\mathsf{F},$$
$$\mathsf{F} = \mathsf{H}\cdot\mathsf{H}, \quad \mathsf{F} = \mathsf{H}\cdot\mathsf{F}, \quad \mathsf{F} = \mathsf{F}\cdot\mathsf{H}, \quad \mathsf{F} = \mathsf{F}\cdot\mathsf{F}.$$

All of them except one have already appeared in the matrix multiplication procedure of the *weakly admissible case*, and the only one that is new is $\mathsf{F} = \mathsf{H}\cdot\mathsf{H}$. This new kind of product is implemented using the randomized SVD algorithm proposed recently in [28, 32] for numerically low-rank matrices. The main idea of this randomized algorithm is to capture the column (or row) space of the matrix by multiplying the matrix with a small number of Gaussian random test vectors. Results from random matrix theory guarantee that the column space of the product matrix approximates accurately the span of all dominant singular vectors of the original (numerically low-rank) matrix. Since the product matrix has far fewer columns, applying singular value decompositions to it gives rise to an accurate and efficient way to approximate the SVD of the original matrix. Notice that this randomized approach only requires a routine to apply the original matrix to an arbitrary vector; everything else is just standard numerical linear algebra. In our setting, applying $\mathsf{H}\cdot\mathsf{H}$ to a vector is simply equal to two hmatvec operations.

2.4 Approximate Inversion and Preconditioner

Let us denote the approximations of S_m and T_m in the hierarchical matrix representation by $\widetilde{S}_m$ and $\widetilde{T}_m$, respectively. The construction of the approximate LDL^{T} factorization of A takes the following steps:

Algorithm 2.5. Construction of the approximate sweeping factorization of A in the hierarchical matrix framework.

1: $\widetilde{S}_1 = A_{1,1}$ and $\widetilde{T}_1 = \mathsf{hinv}(\widetilde{S}_1)$.
2: **for** $m = 2, \ldots, n$ **do**
3: $\quad \widetilde{S}_m = \mathsf{hsub}(A_{m,m}, \mathsf{hdiagmul}(A_{m,m-1}, \mathsf{hdiagmul}(\widetilde{T}_{m-1}, A_{m-1,m})))$ and
$\quad\quad \widetilde{T}_m = \mathsf{hinv}(\widetilde{S}_m)$.
4: **end for**

The cost of Algorithm 2.5 is $O(R^2 n^2 \log^2 n) = O(R^2 N \log^2 N)$. The computation of $u \approx A^{-1} f$ using this approximate factorization is summarized as follows.

Algorithm 2.6. Computation of $u \approx A^{-1} f$ using the approximate sweeping factorization of A in the hierarchical matrix framework.

1: **for** $m = 1, \ldots, n$ **do**
2: $\quad u_m = f_m$
3: **end for**
4: **for** $m = 1, \ldots, n-1$ **do**
5: $\quad u_{m+1} = u_{m+1} - A_{m+1,m} \cdot \mathsf{hmatvec}(\widetilde{T}_m, u_m)$
6: **end for**
7: **for** $m = 1, \ldots, n$ **do**
8: $\quad u_m = \mathsf{hmatvec}(\widetilde{T}_m, u_m)$

end for
for $m = n-1, \ldots, 1$ **do**
  $u_m = u_m - \mathsf{hmatvec}(\widetilde{T}_m, A_{m,m+1}u_{m+1})$
end for

The cost of Algorithm 2.6 is $O(Rn^2 \log n) = O(RN \log N)$. Algorithm 2.6 defines an operator

$$M : f = (f_1^\mathsf{T}, f_2^\mathsf{T}, \ldots, f_n^\mathsf{T})^\mathsf{T} \to u = (u_1^\mathsf{T}, u_2^\mathsf{T}, \ldots, u_n^\mathsf{T})^\mathsf{T},$$

which is an approximate inverse of the discrete Helmholtz operator A. When the threshold ε is set to be sufficiently small, M can be used directly as the inverse of A, and u can be taken as the solution. However, a small ε-value means that the rank R of the low-rank factorized form needs to be fairly large, thus resulting in large storage and computation cost. On the other hand, when R is kept rather small, Algorithms 2.5 and 2.6 become efficient both in terms of storage and time. Though the resulting M is not accurate enough as the inverse of A, it serves as an excellent preconditioner. Therefore, we solve the preconditioner system

$$MAu = Mf$$

using iterative solvers such as GMRES and TFQMR [41, 42]. Since the cost of applying M to any vector is $O(RN \log N)$, the total cost of the iterative solver is $O(N_I RN \log N)$, where N_I is the number of iterations. The numerical results in Section 3 demonstrate that N_I is in practice very small, thus resulting in an algorithm for almost linear complexity.

Theorem 2.3 shows that in the constant-coefficient case the hierarchical matrix representation of T_m is accurate. Therefore, the preconditioner M well approximates the inverse of A, and the number of iterations N_I is expected to be small. The numerical results in Section 3 demonstrate that N_I is also small for a general velocity field such as found in converging lenses, wave guides, and random media. Here we provide a heuristic explanation for this phenomena. For the variable-coefficient case, the numerical rank of the off-diagonal blocks of T_m can potentially increase mainly due to the *turning rays*, i.e., the rays that leave the m^{th} layer downward, then travel horizontally in x_1-direction, and finally come upward back to the m^{th} layer. The interactions related to turning rays are difficult to capture in the hierarchical matrix representation of T_m if R is small. However, the iterative solver addresses this interaction in several steps as follows: the downward part of the ray is processed by a first few sweeps, the horizontal part is then captured by the T_m matrix of the next sweep, and finally the upward part of the ray is processed by a couple of extra sweeps.

In the presentation of the sweeping preconditioner, we choose the sweeping direction to be in the positive x_2-direction. It is clear that sweeping along either one of the other three directions also gives a valid but slightly different sweeping preconditioner. Due to the variations in the velocity field and, more precisely,

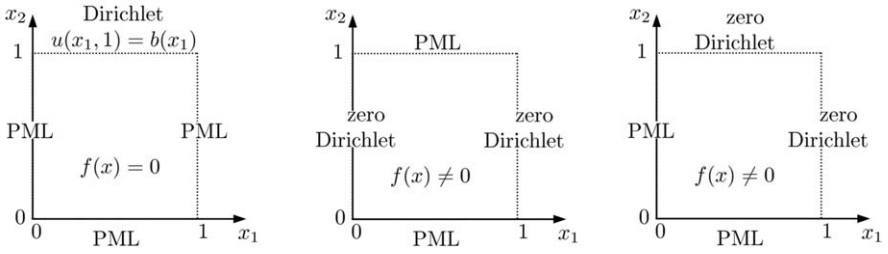


FIGURE 2.5. Mixed boundary conditions. Left: Depth extrapolation problem in seismology. Middle and right: Problems with a partly zero Dirichlet boundary condition and nonzero $f(x)$.

the existence of the turning rays, a carefully selected sweeping direction can often result in a significantly smaller number of GMRES iterations than the other directions. We will give one numerical example to demonstrate this in Section 3.

2.5 Other Boundary Conditions

So far, we have discussed the case with the Sommerfeld boundary condition specified over the whole boundary. From the above discussion, it is clear that the success of the preconditioner relies exclusively on the fact that S_m and T_m are compressible. For many other boundary conditions, the matrices S_m and T_m also have this property, as long as the Helmholtz problem is not close to resonance. Here, we mention three representative cases.

In the first case (see Figure 2.5 (left)), the PML boundary condition at $x_2 = 1$ is replaced with a Dirichlet boundary condition $u(x_1, 1) = b(x_1)$ and f is equal to 0. This problem is known as depth extrapolation [5, 35] in reflection seismology. The proposed algorithm proceeds exactly the same way; the only modification is that the boundary condition $b(x_1)$ is transformed into an appropriate forcing term at the last layer of unknowns (i.e., the index set P_n). We would like to mention that in depth extrapolation the hierarchical matrix representation is used in [43] to approximate 1D spectral projectors.

In the second case, the zero boundary condition is mixed with the PML condition. In Figure 2.5 (middle)) the zero Dirichlet boundary condition is specified on $x_1 = 0$ and $x_1 = 1$. The matrix T_m then corresponds to the restriction (to an edge) of the Green's function of the discrete Helmholtz operator in a half-strip. By using the imaging method also in the x_1-direction, one can show that the ranks of the off-diagonal blocks are bounded by $O(\log \omega |\log \varepsilon|^2)$ with a slightly larger constant due to the mirror images. In Figure 2.5 (right)), the zero Dirichlet boundary condition is specified on $x_1 = 1$ and $x_2 = 1$; here T_m corresponds to the restriction of the Green's function of the discrete Helmholtz operator in a quadrant in this case.

Finally, the PML boundary condition is by no means the only approximation to the Sommerfeld condition. As the essential requirement is that the problem should not be close to resonance (i.e., a wave packet escapes the domain without spending

too much time inside), the sweeping preconditioner should work with any reasonable approximations to the Sommerfeld boundary condition such as absorbing boundary conditions (ABCs) [14, 15] and damping/sponge layers. We focus on the PML due to its simplicity, its low nonphysical reflections, and the symmetry of its discrete system.

3 Numerical Results in 2D

In this section, we present several numerical examples to illustrate the properties of the sweeping preconditioner described in Section 2. The implementation is done in C++ and the results in this section are obtained on a computer with a 2.6-GHz CPU. The GMRES method is used as the iterative solver with relative residue tolerance set to 10^{-3}.

3.1 PML

The numerical examples in this section have the PML boundary condition specified at all sides.

Dependence on ω

First, we study how the sweeping preconditioner behaves when ω varies. Consider three velocity fields in the domain $(0,1)^2$:

(1) a converging lens with a Gaussian profile at the center of the domain (see Figure 3.1(a)),
(2) a vertical waveguide with a Gaussian cross section (see Figure 3.1(b)), and
(3) a random velocity field (see Figure 3.1(c)).

For each velocity field, we perform two tests with different external forces $f(x)$.

(1) $f(x)$ is a Gaussian point source located at $(x_1, x_2) = (0.5, 0.125)$. The response of this forcing term generates circular waves propagating at all directions. Due to the variations of the velocity field, the circular waves should bend, form caustics, and intersect.
(2) $f(x)$ is a Gaussian wave packet with a wavelength comparable to the typical wavelength of the Helmholtz equation. This packet is centered at $(x_1, x_2) = (0.125, 0.125)$ and points in the $(1,1)$ direction. The response of this forcing term generates a Gaussian beam initially pointing in the $(1,1)$ direction. Due to the variations of the velocity field, this Gaussian beam should bend and scatter.

For each velocity field, we discretize with $q = 8$ points per wavelength and perform calculations for $\frac{\omega}{2\pi} = 16, 32, \ldots, 256$. Therefore, the number of points for each dimension is $n = 8 \times \frac{\omega}{2\pi} = 128, 256, \ldots, 2048$. The strongly admissible case is used in the implementation of the hierarchical matrix representation. Recall that R is the rank of the off-diagonal blocks in the hierarchical matrix, and we fix it to be a uniform constant 2. In all tests, the sweeping direction is bottom-up from $x_2 = 0$ to $x_2 = 1$.

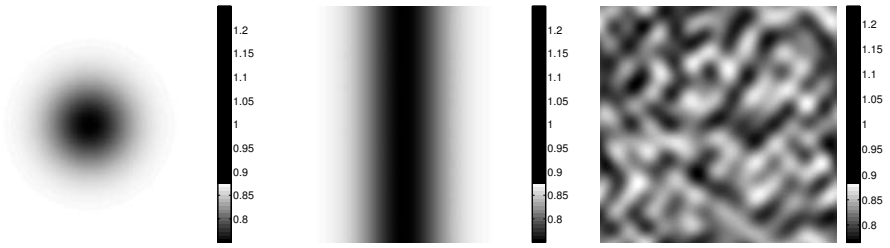


FIGURE 3.1. Test velocity fields.

The results of the first velocity field are summarized in Table 3.1. T_{setup} denotes the time used to construct the preconditioner in seconds. For each external force, N_{iter} is the number of iterations of the preconditioned GMRES solver and T_{solve} is the solution time. When n doubles, N quadruples and the setup cost T_{setup} increases by a factor of 5 or 6, which is consistent with the $O(N \log^2 N)$ complexity of Algorithm 2.5. A remarkable feature of the sweeping preconditioner is that the number of iterations is extremely small. In fact, in all cases, the preconditioned GMRES solver converges in less than three iterations. As a result of the constant iteration number, the solution time increases by a factor of 4 or 5 when N quadruples, which is also consistent with the $O(N \log N)$ complexity of Algorithm 2.6. Finally, we would like to point out that our algorithm is extremely efficient: for a problem with $N = n^2 = 2048^2$ unknowns, the solution time is only about 30 seconds.

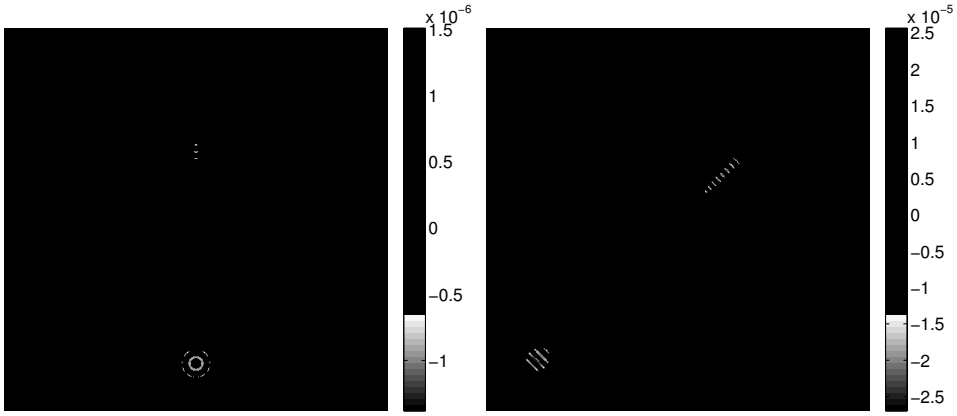
The results of the second and third velocity fields are summarized in Tables 3.2 and 3.3, respectively. The behaviors of these tests are similar to that for the first velocity field. In all cases, the GMRES solver converges in less than five iterations when combined with the sweeping preconditioner.

Dependence on q

Next, we study how the sweeping preconditioner behaves when the number of discretization points per wavelength q varies. We fix $\frac{\omega}{2\pi}$ at 32 and R at 2, and let q be $8, 16, \ldots, 64$. In the following tests, R is again equal to 2. The sweeping direction is bottom-up from $x_2 = 0$ to $x_2 = 1$. The test results for the three velocity fields are summarized in Tables 3.4, 3.5, and 3.6, respectively. These results again show that the number of iterations remains extremely small and the overall solution time scales roughly linearly with respect to the number of unknowns.

Dependence on Sweeping Direction

Finally, we study how the sweeping directions affect the convergence rate of the GMRES algorithm. The velocity field is given by $c(x_1, x_2) = \frac{1}{2} + x_2$, and the external force is a a narrow Gaussian point source centered at $(x_1, x_2) = (0.125, 0.5)$.



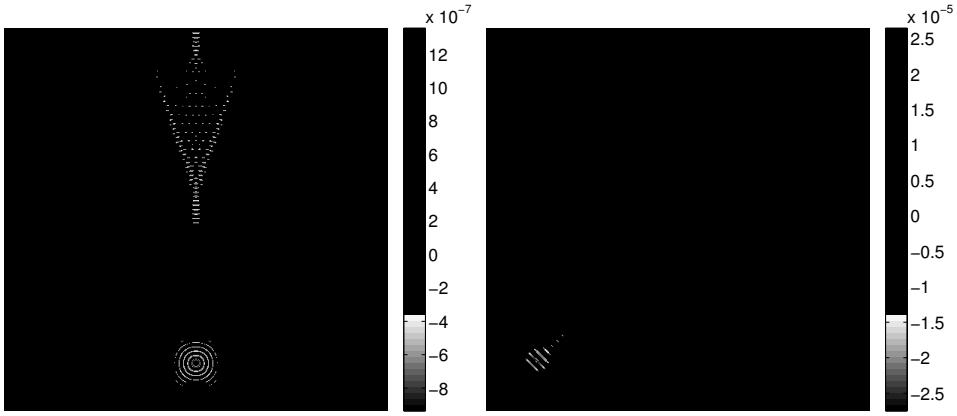
					Test 1		Test 2	
$\omega/(2\pi)$	q	$N = n^2$	R	T_{setup}	N_{iter}	T_{solve}	N_{iter}	T_{solve}
16	8	128^2	2	6.50e–01	2	5.00e–02	2	5.00e–02
32	8	256^2	2	5.05e+00	2	2.50e–01	2	2.50e–01
64	8	512^2	2	3.44e+01	3	1.45e+00	3	1.42e+00
128	8	1024^2	2	2.16e+02	3	7.37e+00	3	7.36e+00
256	8	2048^2	2	1.24e+03	3	3.31e+01	3	3.28e+01

TABLE 3.1. Results of velocity field 1 for different ω. Top: Solutions for two external forces with $\frac{\omega}{2\pi} = 64$. Bottom: Results for different ω.

Two sweeping directions are tested here: the first one sweeps in the positive x_2-direction while the second sweeps in the negative x_2-direction. For the first sweeping direction, the matrix T_m approximates the Green's function of the lower half-space $(-\infty, \infty) \times (-\infty, mh)$. Since the velocity field decreases in the negative x_2-direction, in a geometric optics argument the rays emanating from the $x_2 = mh$ plane do not travel back to the same plane. Therefore, we expect the numerical rank of the off-diagonal blocks of T_m to be low, the preconditioner to be quite accurate, and the number of iterations to be small. The geometric theory of diffraction indicates that the coupling between points on the plane $x_2 = mh$ is via exponentially decaying creeping rays and thus very weak.

For the second sweeping direction, the matrix T_m approximates the Green's function of the upper half-space $(-\infty, \infty) \times (1 - mh, \infty)$. Since the velocity field increases in the positive x_2-direction, the rays emanating from $x_2 = 1 - mh$ can shoot back to the same plane. As a result, the hierarchical matrix representation of T_m would incur a larger error for the same R-value and the number of iterations would become larger.

Table 3.7 reports the results of these two sweeping directions for different ω-values. As expected by the above argument, the number of iterations for the first sweeping preconditioner (in the positive x_2-direction) remains very small, while



					Test 1		Test 2	
$\omega/(2\pi)$	q	$N = n^2$	R	T_{setup}	N_{iter}	T_{solve}	N_{iter}	T_{solve}
16	8	128^2	2	6.70e–01	2	5.00e–02	2	6.00e–02
32	8	256^2	2	4.97e+00	2	2.30e–01	2	2.30e–01
64	8	512^2	2	3.43e+01	3	1.39e+00	3	1.39e+00
128	8	1024^2	2	2.13e+02	4	8.43e+00	4	8.38e+00
256	8	2048^2	2	1.25e+03	5	4.65e+01	4	3.93e+01

TABLE 3.2. Results of velocity field 2 for different ω. Top: Solutions for two external forces with $\frac{\omega}{2\pi} = 64$. Bottom: Results for different ω.

the number of iterations for the second one (in the negative x_2-direction) increases slightly with N.

3.2 Other Boundary Conditions

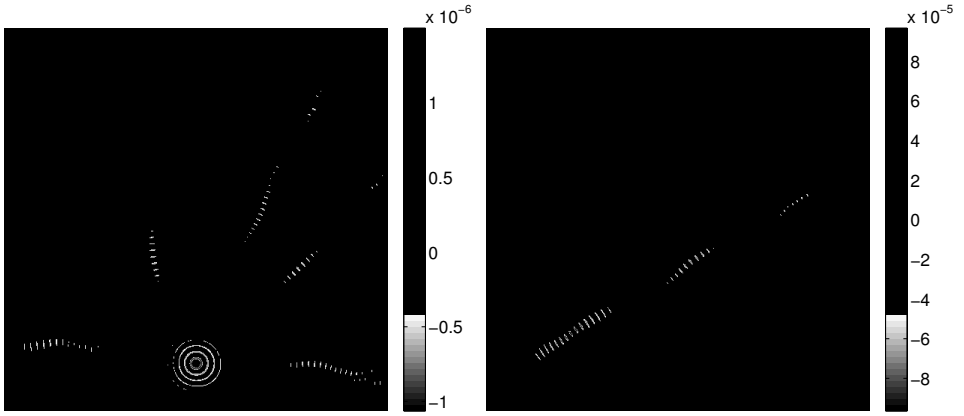
Here we report numerical examples with different boundary conditions.

Depth Extrapolation

In this example (see Figure 2.5 (left)), the velocity field is a vertical wave guide. We specify a Dirichlet boundary condition $u(x_1, 1) = b(x_1)$ at the top edge $x_2 = 1$ and the PML at the other three edges. This is the depth extrapolation problem in reflection seismology and we report the results of two tests:

(1) $b(x_1) = 1$. This corresponds to a plane wave entering the wave guide. The center part of the plane wave should start to bend and eventually form multiple caustics.
(2) $b(x_1) = \exp(i\frac{\omega}{2}x_1)$. This corresponds to a slant wave entering the wave guide.

The sweeping direction is bottom-up from $x_2 = 0$ to $x_2 = 1$; the results are summarized in Table 3.8. The running time again closely follows the analytical estimate, and the number of GMRES iterations are bounded by 4.



					Test 1		Test 2	
$\omega/(2\pi)$	q	$N = n^2$	R	T_{setup}	N_{iter}	T_{solve}	N_{iter}	T_{solve}
16	8	128^2	2	6.50e–01	2	5.00e–02	2	5.00e–02
32	8	256^2	2	5.10e+00	2	2.50e–01	3	3.00e–01
64	8	512^2	2	3.48e+01	3	1.49e+00	3	1.48e+00
128	8	1024^2	2	2.16e+02	4	8.99e+00	3	7.37e+00
256	8	2048^2	2	1.26e+03	5	4.64e+01	3	3.25e+01

TABLE 3.3. Results of velocity field 3 for different ω. Top: Solutions for two external forces with $\frac{\omega}{2\pi} = 64$. Bottom: Results for different ω.

					Test 1		Test 2	
$\omega/(2\pi)$	q	$N = n^2$	R	T_{setup}	N_{iter}	T_{solve}	N_{iter}	T_{solve}
32	8	256^2	2	4.93e+00	2	2.30e–01	2	2.30e–01
32	16	512^2	2	3.42e+01	2	1.11e+00	2	1.09e+00
32	32	1024^2	2	2.13e+02	2	5.45e+00	2	5.45e+00
32	64	2048^2	2	1.23e+03	2	2.50e+01	2	2.49e+01

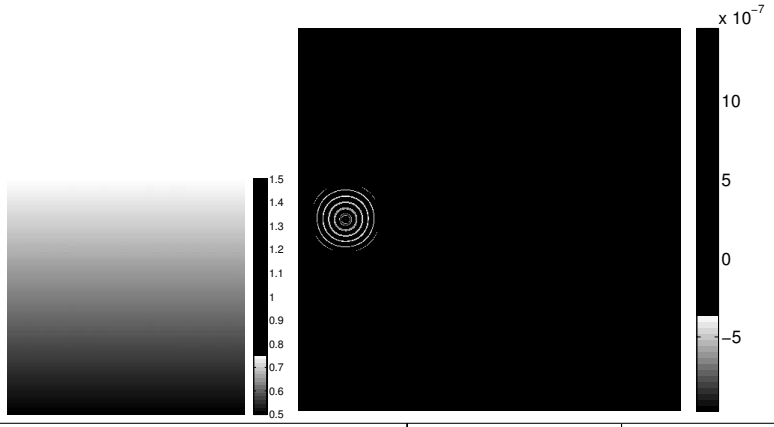
TABLE 3.4. Results of velocity field 1 for different q.

					Test 1		Test 2	
$\omega/(2\pi)$	q	$N = n^2$	R	T_{setup}	N_{iter}	T_{solve}	N_{iter}	T_{solve}
32	8	256^2	2	4.93e+00	2	2.30e–01	2	2.30e–01
32	16	512^2	2	3.42e+01	2	1.11e+00	2	1.09e+00
32	32	1024^2	2	2.13e+02	2	5.45e+00	2	5.37e+00
32	64	2048^2	2	1.23e+03	2	2.50e+01	2	2.49e+01

TABLE 3.5. Results of velocity field 2 for different q.

					Test 1		Test 2	
$\omega/(2\pi)$	q	$N = n^2$	R	T_{setup}	N_{iter}	T_{solve}	N_{iter}	T_{solve}
32	8	256^2	2	5.13e+00	2	2.40e–01	3	3.10e–01
32	16	512^2	2	3.47e+01	2	1.21e+00	2	1.20e+00
32	32	1024^2	2	2.14e+02	2	5.87e+00	2	5.84e+00
32	64	2048^2	2	1.23e+03	2	2.52e+01	2	2.51e+01

TABLE 3.6. Results of velocity field 3 for different q.



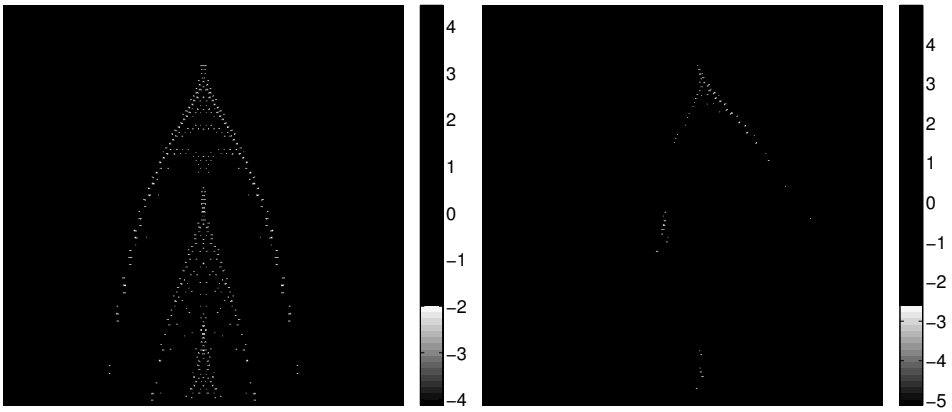
					Positive x_2		Negative x_2	
$\omega/(2\pi)$	q	$N = n^2$	R	T_{setup}	N_{iter}	T_{solve}	N_{iter}	T_{solve}
16	8	128^2	2	6.80e–01	1	4.00e–02	2	5.00e–02
32	8	256^2	2	4.95e+00	2	2.50e–01	3	3.00e–01
64	8	512^2	2	3.40e+01	2	1.13e+00	4	1.86e+00
128	8	1024^2	2	2.14e+02	2	5.82e+00	6	1.21e+01
256	8	2048^2	2	1.25e+03	2	2.64e+01	6	5.49e+01

TABLE 3.7. Results of the positive and negative x_2 sweeping directions. Top row: the velocity field (left) and the solution for the external force (right). Bottom row: results for different ω.

Mixed PML-Dirichlet Boundary Condition

Here the velocity field $c(x)$ is equal to the constant 1 and we perform two tests with mixed boundary conditions.

(1) In the first test (see Figure 2.5 (middle)), we specify the zero Dirichlet boundary condition at $x_1 = 0$ and $x_1 = 1$ and the PML condition at the other two sides. The external force $f(x)$ is a Gaussian wave packet with a wavelength comparable to the typical wavelength of the Helmholtz equation. This packet centers at $(x_1, x_2) = (0.5, 0.125)$ and points to



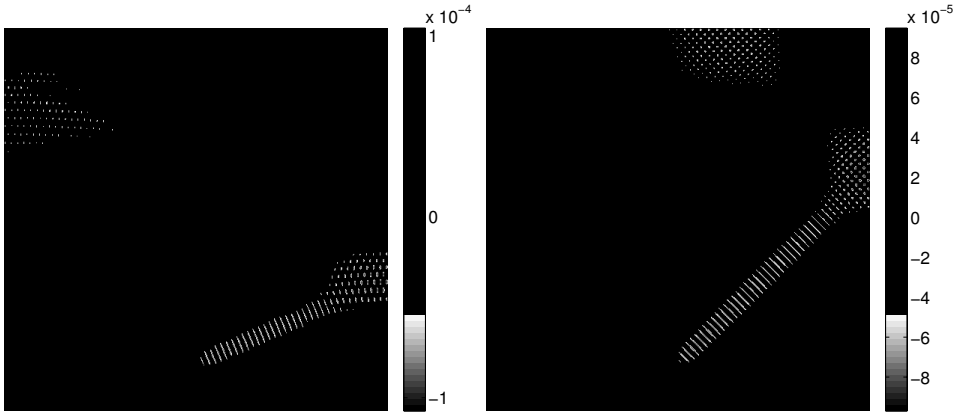
					Test 1		Test 2	
$\omega/(2\pi)$	q	$N = n^2$	R	T_{setup}	N_{iter}	T_{solve}	N_{iter}	T_{solve}
16	8	128^2	2	6.60e–01	2	4.00e–02	2	4.00e–02
32	8	256^2	2	5.07e+00	3	3.20e–01	3	3.00e–01
64	8	512^2	2	3.45e+01	3	1.48e+00	3	1.46e+00
128	8	1024^2	2	2.15e+02	3	7.29e+00	3	7.30e+00
256	8	2048^2	2	1.25e+03	4	3.92e+01	4	3.94e+01

TABLE 3.8. Results of the depth-stepping example for different ω. Top: Solutions for two test cases with $\frac{\omega}{2\pi} = 64$. Bottom: Results for different ω.

the direction $(\cos(\pi/8), \sin(\pi/8))$. The Gaussian beam generated by this forcing term should bounce back from the edge $x_1 = 1$ and then from the edge $x_1 = 0$.

(2) In the second test (see Figure 2.5 (right)), we specify the zero Dirichlet boundary condition at $x_1 = 1$ and $x_2 = 1$ and the PML condition at the other two sides. The external force $f(x)$ is a Gaussian wave packet with a wavelength comparable to the typical wavelength of the Helmholtz equation. This packet centers at $(x_1, x_2) = (0.5, 0.125)$ and points to the direction $(1, 1)$. The Gaussian beam generated by the external force should bounce back from the edge $x_1 = 1$ and then from the edge $x_2 = 1$.

The sweeping direction is bottom-up from $x_2 = 0$ to $x_2 = 1$, and the results of these tests are summarized in Table 3.9. The running time again follows the analytical estimate. In the first test, due to the reason mentioned in Section 2.5, the rank of the off-diagonal blocks of the Schur complement matrices is slightly higher. Hence, with the same R-value the number of iterations is expected to increase slightly. In all cases, the number of GMRES iterations is bounded by 10.



					Test 1		Test 2	
$\omega/(2\pi)$	q	$N = n^2$	R	T_{setup}	N_{iter}	T_{solve}	N_{iter}	T_{solve}
16	8	128^2	2	6.80e–01	2	5.00e–02	2	5.00e–02
32	8	256^2	2	5.00e+00	3	3.10e–01	2	2.50e–01
64	8	512^2	2	3.47e+01	6	2.70e+00	2	1.27e+00
128	8	1024^2	2	2.16e+02	9	1.80e+01	2	6.19e+00
256	8	2048^2	2	1.26e+03	10	8.52e+01	2	2.69e+01

TABLE 3.9. Results of the mixed boundary condition example for different ω. Top: Solutions for two test cases with $\frac{\omega}{2\pi} = 64$. Bottom: Results for different ω.

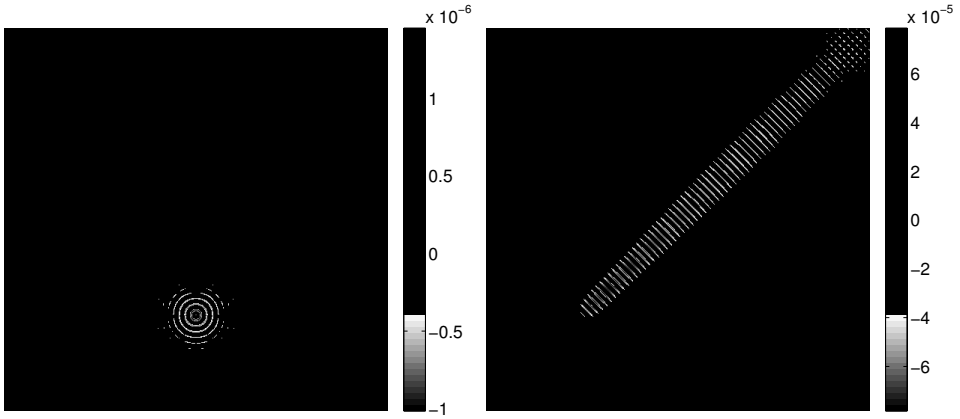
Absorbing Boundary Condition

In the last example, we replace the PML with the second-order absorbing boundary condition (ABC). The velocity field $c(x)$ is taken to be 1, and we perform tests with two different external forces, which are similar to the ones given at the beginning of Section 3.1.

(1) The first external force $f(x)$ is a Gaussian point source located at $(x_1, x_2) = (0.5, 0.25)$.
(2) The second external force $f(x)$ is a Gaussian wave packet with a wavelength comparable to the typical wavelength of the Helmholtz equation. This packet centers at $(x_1, x_2) = (0.25, 0.25)$ and points in the direction $(1, 1)$.

Notice that since the low-order ABCs generate more nonphysical reflections at the domain boundaries, here we move the support of these external forces closer to the center of the computational domain.

Due to the same nonphysical reflections, the discrete Green's function associated with a low-order ABC often has off-diagonal blocks with higher numerical ranks compared to the discrete Green's function associated with the PML. As a result, we let R increase slightly with ω. The sweeping direction is bottom-up



					Test 1		Test 2	
$\omega/(2\pi)$	q	$N = n^2$	R	T_{setup}	N_{iter}	T_{solve}	N_{iter}	T_{solve}
16	8	128^2	2	6.70e–01	7	1.30e–01	6	1.50e–01
32	8	256^2	2	4.98e+00	7	4.80e–01	6	4.30e–01
64	8	512^2	3	5.10e+01	8	3.16e+00	6	2.42e+00
128	8	1024^2	4	4.65e+02	10	2.06e+01	6	1.33e+01
256	8	2048^2	5	3.84e+03	13	1.59e+01	6	8.49e+01

TABLE 3.10. Results of the absorbing boundary condition (ABC) test for different ω. Top: Solutions for two test cases with $\omega/(2\pi) = 64$. Bottom: Results for different ω.

from $x_2 = 0$ to $x_2 = 1$, and the results are summarized in Table 3.10. The setup time grows slightly higher than linear complexity due to the increase of R, and the number of iterations increases roughly logarithmically with respect to ω. In all cases, the number of GMRES iterations is bounded by 13. Overall, the results for the ABC are slightly worse than the ones for the PML, suggesting that, in order for the sweeping preconditioner to work well, it is essential to minimize nonphysical reflections at the domain boundary.

4 Preconditioner in 3D

4.1 Discretization

The computational domain is $D = (0, 1)^3$. Using the same $\sigma(t)$ introduced in (2.1), we define

$$s_1(x_1) = \left(1 + i\frac{\sigma(x_1)}{\omega}\right)^{-1}, \quad s_2(x_2) = \left(1 + i\frac{\sigma(x_2)}{\omega}\right)^{-1},$$
$$s_3(x_3) = \left(1 + i\frac{\sigma(x_3)}{\omega}\right)^{-1}.$$

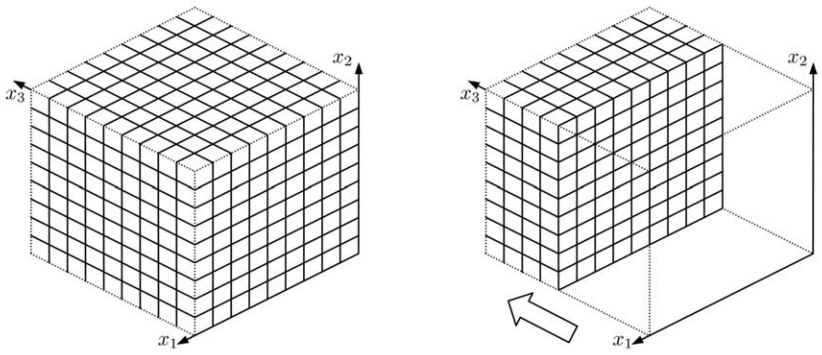


FIGURE 4.1. Left: Discretization grid in 3D. Right: Sweeping order in 3D. The remaining grid shows the unknowns yet to be processed.

The PML replaces ∂_1 with $s_1(x_1)\partial_1$, ∂_2 with $s_2(x_2)\partial_2$, and ∂_3 with $s_3(x_3)\partial_3$. This effectively provides a damping layer of width η near the boundary of $D = (0,1)^3$. The resulting equation is

$$\left((s_1\partial_1)(s_1\partial_1) + (s_2\partial_2)(s_2\partial_2) + (s_3\partial_3)(s_3\partial_3) + \frac{\omega^2}{c^2(x)}\right)u = f, \quad x \in D = [0,1]^3,$$
$$u = 0, \quad x \in \partial D.$$

Without loss of generality, we assume that $f(x)$ is supported inside $[\eta, 1-\eta]^3$ (away from the PML). Dividing the above equation by $s_1 s_2 s_3$ yields

$$\left(\partial_1\left(\frac{s_1}{s_2 s_3}\partial_1\right) + \partial_2\left(\frac{s_2}{s_1 s_3}\partial_2\right) + \partial_3\left(\frac{s_3}{s_1 s_2}\partial_3\right) + \frac{\omega^2}{s_1 s_2 s_3 c^2(x)}\right)u = f.$$

The domain $[0,1]^3$ is discretized with a Cartesian grid with spacing $h = \frac{1}{n+1}$, where n is again assumed to be a power of 2. As we discretize the equation with a few of points per wavelength, the number n of samples in each dimension is proportional to ω. The interior points of this grid are

$$P = \{p_{i,j,k} = (ih, jh, kh) : 1 \le i, j, k \le n\}$$

(see Figure 4.1 (left)), and the total number of points is equal to $N = n^3$.

We denote by $u_{i,j,k}$, $f_{i,j,k}$, and $c_{i,j,k}$ the values of $u(x)$, $f(x)$, and $c(x)$ at point $p_{i,j,k} = (ih, jh, kh)$. Once we discretize the problem at points in P with the seven-point central difference scheme, at $p_{i,j,k} = (ih, jh, kh)$ the resulting equation is

$$\begin{aligned}(4.1)\qquad &\frac{1}{h^2}\left(\frac{s_1}{s_2 s_3}\right)_{i-\frac{1}{2},j,k} u_{i-1,j,k} + \frac{1}{h^2}\left(\frac{s_1}{s_2 s_3}\right)_{i+\frac{1}{2},j,k} u_{i+1,j,k} \\ &+ \frac{1}{h^2}\left(\frac{s_2}{s_1 s_3}\right)_{i,j-\frac{1}{2},k} u_{i,j-1,k} + \frac{1}{h^2}\left(\frac{s_2}{s_1 s_3}\right)_{i,j+\frac{1}{2},k} u_{i,j+1,k} +\end{aligned}$$

$$+\frac{1}{h^2}\left(\frac{s_3}{s_1s_2}\right)_{i,j,k-\frac{1}{2}}u_{i,j,k-1}+\frac{1}{h^2}\left(\frac{s_3}{s_1s_2}\right)_{i,j,j+\frac{1}{2}}u_{i,j,k+1}$$
$$+\left(\frac{\omega^2}{(s_1s_2s_3)_{i,j,k}\cdot c^2_{i,j,k}}-(\cdots)\right)u_{i,j,k}=f_{i,j,k}$$

with $u_{i',j',k'}$ equal to 0 for (i',j',k') that violates $1\le i',j',k'\le n$. Here $(\cdots)$ stands for the sum of the six coefficients appearing in the first six terms. We order $u_{i,j,k}$ by going through the dimensions in order and denote the vector containing all unknowns by

$$u=(u_{1,1,1},u_{2,1,1},\ldots,u_{n,1,1},\ldots,u_{1,n,n},u_{2,n,n},\ldots,u_{n,n,n})^{\mathsf{T}}.$$

Similarly, the $f_{i,j,k}$ are ordered in the same way and the vector f is

$$f=(f_{1,1,1},f_{2,1,1},\ldots,f_{n,1,1},\ldots,f_{1,n,n},f_{2,n,n},\ldots,f_{n,n,n})^{\mathsf{T}}.$$

By denoting the linear operator in (4.1) by A, we obtain a linear system $Au=f$. We further introduce a block version by defining P_m to be the indices in the m^{th} row

$$P_m=\{p_{1,1,m},p_{2,1,m},\ldots,p_{n,n,m}\}$$

and introducing

$$u_m=(u_{1,1,m},u_{2,1,m},\ldots,u_{n,n,m})^{\mathsf{T}},\quad f_m=(f_{1,1,m},f_{2,1,m},\ldots,f_{n,n,m})^{\mathsf{T}}.$$

Then

$$u=(u_1^{\mathsf{T}},u_2^{\mathsf{T}},\ldots,u_n^{\mathsf{T}})^{\mathsf{T}},\quad f=(f_1^{\mathsf{T}},f_2^{\mathsf{T}},\ldots,f_n^{\mathsf{T}})^{\mathsf{T}}.$$

Using this notation, the system $Au=f$ takes the following block tridiagonal form:

$$\begin{pmatrix}A_{1,1}&A_{1,2}&&&\\A_{2,1}&A_{2,2}&\ddots&&\\&\ddots&\ddots&\ddots&\\&&\ddots&\ddots&A_{n-1,n}\\&&&A_{n,n-1}&A_{n,n}\end{pmatrix}\begin{pmatrix}u_1\\u_2\\\vdots\\\vdots\\u_n\end{pmatrix}=\begin{pmatrix}f_1\\f_2\\\vdots\\\vdots\\f_n\end{pmatrix}$$

where each block $A_{i,j}$ is of size $n^2\times n^2$ and $A_{m,m-1}=A_{m-1,m}^{\mathsf{T}}$ are diagonal matrices. Similar to the 2D case, the sweeping factorization eliminates the unknowns face by face, starting from the face next to $x_3=0$ (illustrated in Figure 4.1 (right)). The algorithms for constructing and applying the sweeping factorization are exactly the same as Algorithms 2.1 and 2.2). The matrix $T_m=S_m^{-1}$ is the restriction to $x_3=mh$ of the discrete half-space Green's function with zero boundary condition at $x_3=(m+1)h$. Recall that in the 2D case the off-diagonal blocks of T_m are numerically low-rank. In the 3D case, the rank may be somewhat higher. On the other hand, since we only aim at constructing a preconditioner for the Helmholtz problem, it is still reasonable to introduce a hierarchical structure on the unknowns on the face $x_3=mh$ and use the hierarchical matrix framework to approximate T_m and S_m.

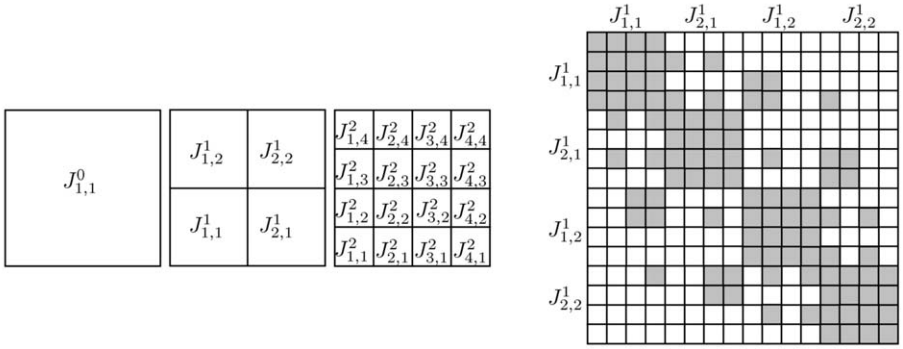


FIGURE 4.2. Hierarchical matrix representation. Left: hierarchical decomposition of the index set J for each layer. Right: Induced partitioning of the matrix T_m in the strongly admissible case. Blocks in white are stored in low-rank factorized form. Blocks in gray are stored densely.

4.2 Hierarchical Matrix Representation

At the m^{th} layer for any fixed m, we build a hierarchical structure for the grid points in P_m through bisections in both the x_1- and x_2-directions. At the top level (level 0), we set

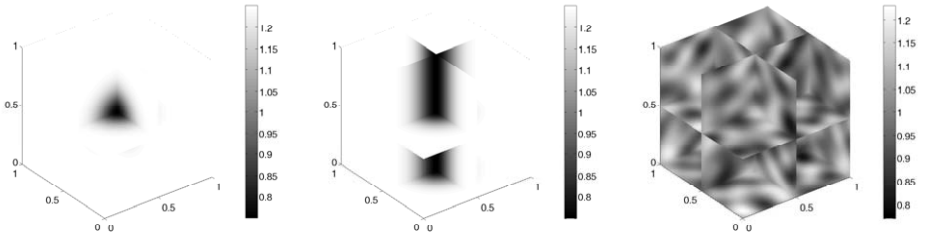
$$J^0_{11} = P_m.$$

At level ℓ, there are $2^\ell \times 2^\ell$ index sets J^ℓ_{ij} with $i, j = 1, \ldots, 2^\ell$:

$$J^\ell_{ij} = \{p_{s,t,m} : (i-1)\cdot n/2^\ell + 1 \le s \le i \cdot n/2^\ell, (j-1)\cdot n/2^\ell + 1 \le t \le j \cdot n/2^\ell\}.$$

The bisection is stopped when each set J^ℓ_{ij} contains only a small number of indices. Hence, the total number of levels L is equal to $\log_2 n - O(1)$. This hierarchical partition is illustrated in Figure 4.2 (left).

We write $G(J^\ell_{ij}, J^\ell_{i'j'})$ (the restriction of a matrix G to J^ℓ_{ij} and $J^\ell_{i'j'}$) as $G^\ell_{ij,i'j'}$. The strongly admissible case is used here, and two index sets J^ℓ_{ij} and $J^\ell_{i'j'}$ on the same level ℓ are considered well separated from each other if $\max(|i - i'|, |j - j'|) > 1$. Recall that the interaction list of J^ℓ_{ij} is the set of all index sets $J^\ell_{i'j'}$ such that J^ℓ_{ij} is well separated from $J^\ell_{i'j'}$ but J^ℓ_{ij}'s parent is not well separated from $J^\ell_{i'j'}$'s parent. When J^ℓ_{ij} and $J^\ell_{i'j'}$ are well-separated from each other, the numerical rank of their interaction $G^\ell_{ij,i'j'}$ is of order $O(n/2^\ell)$. As the number of indices in J^ℓ_{ij} and $J^\ell_{i'j'}$ is equal to $(n/2^\ell)^2$, the numerical rank scales like the square root of the number of indices in each set. Therefore, it is still favorable to store the interaction $G^\ell_{ij,i'j'}$ in a factorized form. In principle, the rank R of the factorized form should scale like $O(n/2^\ell)$. However, since the construction cost of the approximate sweeping factorization scales like $O(R^2 n^3 \log^2 n) = O(R^2 N \log^2 N)$, following this scaling can be rather costly in practice. Instead, we choose R to be



(a) (b) (c)

FIGURE 5.1. Test velocity fields. For each velocity field, the cross sections at $x_1 = 0.5$, $x_2 = 0.5$, and $x_3 = 0.5$ are shown.

a rather small constant, since our goal is only to construct a preconditioner. An illustration of this hierarchical representation is given in Figure 4.2 (right).

Once the details of the hierarchical matrix representation are determined, the construction of the approximate LDL^{T} factorization and the application of its inverse take the same form as Algorithms 2.5 and 2.6, respectively. The operator

$$M : f = (f_1^{\mathsf{T}}, f_2^{\mathsf{T}}, \ldots, f_n^{\mathsf{T}})^{\mathsf{T}} \to u = (u_1^{\mathsf{T}}, u_2^{\mathsf{T}}, \ldots, u_n^{\mathsf{T}})^{\mathsf{T}}$$

defined by Algorithm 2.6 is an approximate inverse and a good preconditioner of the discrete Helmholtz operator A. Therefore, we solve the preconditioner system

$$MAu = Mf$$

using GMRES or TFQMR. Since the cost of applying M to an arbitrary vector is $O(Rn^3 \log n) = O(RN \log N)$, the total cost is

$$O(N_I R n^3 \log n) = O(N_I RN \log N),$$

where N_I is the number of iterations. The numerical results in Section 5 demonstrate that N_I and R are in practice rather small.

5 Numerical Results in 3D

In this section, we present several numerical examples to illustrate the properties of the sweeping preconditioner described in Section 4. We use the GMRES method as the iterative solver with relative residue tolerance set to 10^{-3}. The examples in this section have the PML boundary condition specified at all sides.

We consider three velocity fields in the domain $[0, 1]^3$:

(1) a converging lens with a Gaussian profile at the center of the domain (see Figure 5.1(a)),
(2) a vertical waveguide with Gaussian cross section (see Figure 5.1(b)), and
(3) a random velocity field (see Figure 5.1(c)).

For each problem, we perform two tests with different external forces $f(x)$.

(1) $f(x)$ is a Gaussian point source that is located at the point $(x_1, x_2, x_3) = (0.50, 0.50, 0.25)$. The response of this forcing term generates spherical waves propagating in all directions. Due to the variations of the velocity field, the circular waves should bend and form caustics.
(2) $f(x)$ is a Gaussian wave packet whose wavelength is comparable to the typical wavelength of the domain. This packet is centered at $(x_1, x_2, x_3) = (0.50, 0.25, 0.25)$ and points in the direction $(0, 1, 1)$. The response of this forcing term generates a Gaussian beam initially pointing in the direction $(0, 1, 1)$.

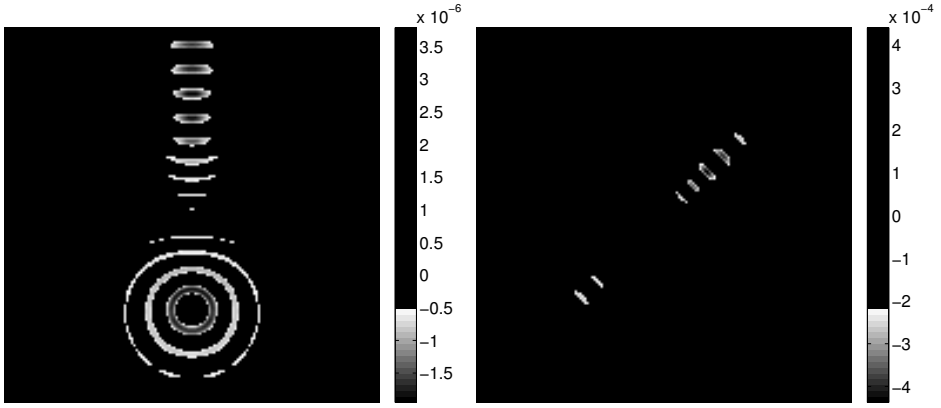
For each velocity field, we discretize with $q = 8$ points per wavelength and perform calculations for $\frac{\omega}{2\pi}$ equal to $5, 10, 20$. Hence, in these tests the number of points in each dimension is $n = 40, 80, 160$. Recall that R is the rank of the factorized form of the hierarchical matrix representation. It is clear from the discussion of Section 4.2 that the value of R should grow with ω (and n). Here, we choose $R = 2, 3, 4$ for $\omega = 5, 10, 20$, respectively. The sweeping direction is bottom-up from $x_3 = 0$ to $x_3 = 1$.

The results of the first velocity field are reported in Table 5.1. The two plots show the solutions of the two external forces on an (x_0, x_2)-plane near $x_1 = \frac{1}{2}$. T_{setup} is the time used to construct the preconditioner in seconds, N_{iter} is the number of iterations of the preconditioned GMRES solver, and T_{solve} is the solution time. The analysis in Section 4.2 shows that the setup time scales like $O(R^2 n^3 \log^2 n) = O(R^2 N \log^2 N)$. When ω grows from 5 to 20, since R increases from 2 to 4, T_{setup} increases by a factor of 20 times each time ω doubles. Though the setup cost grows significantly faster than the linear scaling $O(N)$, it is still much better than the $O(N^2)$ scaling of the multifrontal method. Notice that the number of iterations of the sweeping preconditioner is extremely small. In fact, in all cases, the GMRES solver converges in at most seven iterations. Finally, we would like to point out that our algorithm is quite efficient once the preconditioner is constructed: for the case with $\frac{\omega}{2\pi} = 20$ with more than four million unknowns, the solution time is only about three minutes.

The results of the second and the third velocity fields are reported in Tables 5.2 and 5.3, respectively. In all cases, the GMRES solver converges in at most five iterations when combined with the sweeping preconditioner.

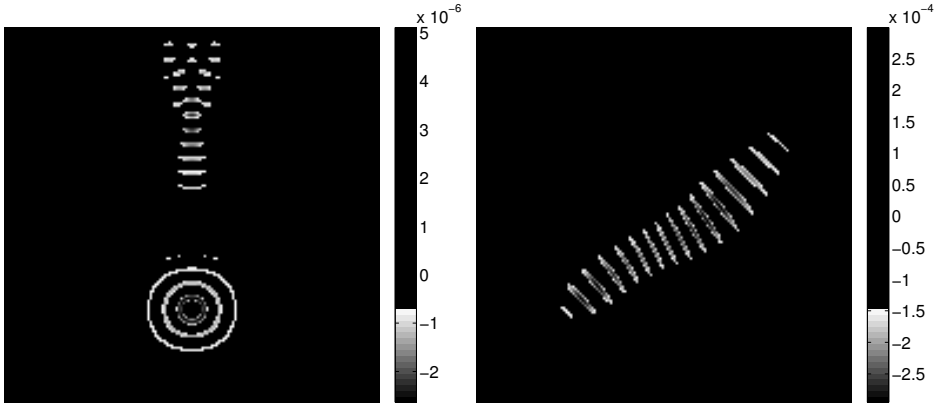
6 Conclusion and Future Work

In this paper, we have proposed a sweeping preconditioner for the iterative solution of variable-coefficient Helmholtz equations in two and three dimensions. The construction of the preconditioner is based on an approximate block LDL^{T} factorization that eliminates the unknowns layer by layer starting from an absorbing



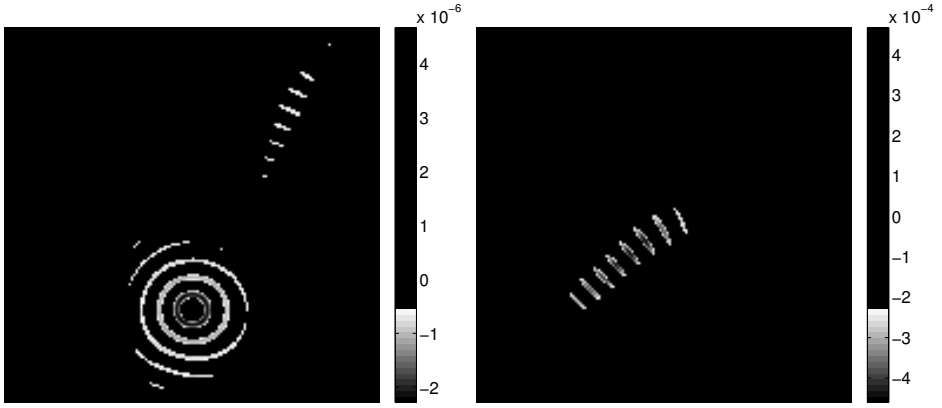
					Test 1		Test 2	
$\omega/(2\pi)$	q	$N = n^3$	R	T_{setup}	N_{iter}	T_{solve}	N_{iter}	T_{solve}
5	8	40^3	2	8.99e+01	3	7.10e–01	3	7.20e–01
10	8	80^3	3	2.30e+03	7	1.87e+01	5	1.40e+01
20	8	160^3	4	4.73e+04	6	1.90e+02	5	1.61e+02

TABLE 5.1. Results of velocity field 1 for different ω. Top: Solutions for two external forces with $\frac{\omega}{2\pi} = 20$ on an (x_0, x_2)-plane near $x_1 = 0.5$. Bottom: Results for different ω.



					Test 1		Test 2	
$\omega/(2\pi)$	q	$N = n^3$	R	T_{setup}	N_{iter}	T_{solve}	N_{iter}	T_{solve}
5	8	40^3	2	8.95e+01	3	7.10e–01	3	7.00e–01
10	8	80^3	3	2.35e+03	5	1.40e+01	3	9.38e+00
20	8	160^3	4	4.73e+04	4	1.38e+02	4	1.34e+02

TABLE 5.2. Results of velocity field 2 for different ω. Top: Solutions for two external forces with $\frac{\omega}{2\pi} = 20$ on an (x_0, x_2)-plane near $x_1 = 0.5$. Bottom: Results for different ω.



					Test 1		Test 2	
$\omega/(2\pi)$	q	$N = n^3$	R	T_{setup}	N_{iter}	T_{solve}	N_{iter}	T_{solve}
5	8	40^3	2	9.00e+01	3	7.20e–01	3	7.20e–01
10	8	80^3	3	2.37e+03	4	1.22e+01	3	9.90e+00
20	8	160^3	4	4.74e+04	4	1.37e+02	3	1.07e+02

TABLE 5.3. Results of velocity field 3 for different ω. Top: Solutions for two external forces with $\frac{\omega}{2\pi} = 20$ on an (x_0, x_2)-plane near $x_1 = 0.5$. Bottom: Results for different ω.

layer. By representing and manipulating the intermediate Schur complement matrices in the hierarchical matrix framework, we have obtained preconditioners with almost linear cost. Numerical examples demonstrate that, when combined with standard iterative solvers, these new preconditioners result in almost ω-independent iteration numbers.

Some questions remain open. First, in the 2D case, we have proved the compressibility result under the constant-coefficient case. A natural question is to what extent this is still true for a general velocity field.

The hierarchical matrix representation may not be very accurate for the Schur complement matrices in 3D, since some high-rank off-diagonal blocks are stored in a low-rank factorized form. Yet our algorithm works well with very small iteration numbers. It is important to understand why this is the case and also to investigate whether other matrix representations would be able to provide more accurate approximations for T_m.

The memory space required by the sweeping preconditioners is linear with respect to the number of unknowns. However, the prefactor is higher compared to the shifted Laplacian preconditioners and the ILU preconditioners. Most of the memory space is in fact used to store the diagonal part of T_m, which corresponds to the local part of the half-space Green's function. One improvement is to use the

asymptotic formula of the Green's function to represent the local part of T_m analytically; this can eliminate the need for storing the diagonal part of the hierarchical matrices.

The matrix representation used here is often referred to as the H^1 form of the hierarchical matrix algebra. More efficient and sophisticated versions are the uniform H^1 form and the H^2 form. For our problem, Algorithm 2.5 requires the matrices to be represented in the H^1 form since it uses the matrix inversion procedure. However, Algorithm 2.6 for applying the sweeping preconditioner can potentially speed up dramatically when the H^2 form is used.

We have chosen the PML for the numerical implementation of the Sommerfeld condition. Many other boundary conditions are available and commonly used. The sweeping approach should work for these boundary conditions, as we have briefly demonstrated for the second-order ABC. The design and implementation of these other boundary conditions should minimize nonphysical reflections in order for the sweeping preconditioner to perform well.

The second-order central difference scheme is used to discretize the Helmholtz equation in this paper. We would like to investigate other more accurate stencils and other types of discretizations such as h/p finite elements, spectral elements, and discontinuous Galerkin methods.

Since high-frequency fields typically oscillate rapidly on a similar scale throughout the computational domain, uniform grids are quite common. There are, however, situations where unstructured grids would be natural. The sweeping approach and more general hierarchical matrix representations can also be used in this context. The challenge here is to maintain compatibility between the matrix representation and the geometry as one sweeps through the computational domain. In a second paper [19] another variant of sweeping preconditioning is presented, which is more flexible with respect to unstructured and adaptive grids.

The sequential nature of the sweeping approach complicates parallelization of the algorithm. One possibility is to use a parallel hierarchical matrix representation for each layer, which would parallelize an inner part of the algorithm. Another technique would leverage the idea of domain decomposition and use the sweeping preconditioner within each subdomain. The subdomains should then be coupled with absorbing boundary conditions.

The Helmholtz equation is only the simplest example of time-harmonic wave equations. Other cases include various elasticity equations and Maxwell equations. For these more complicated systems, multiple wave numbers coexist even for the constant-coefficient case. The basic idea of the sweeping preconditioner should apply but the details need to be worked out.

Acknowledgment. B.E. is partially supported by National Science Foundation grants DMS-0714612 and DMS-1016577. L.Y. is partially supported by National

Science Foundation CAREER Award DMS-0846501, National Science Foundation Grant DMS-1016577, and an Alfred P. Sloan fellowship. The authors thank Leszek Demkowicz and Laurent Demanet for discussions and suggestions.

Bibliography

[1] Bayliss, A.; Goldstein, C. I.; Turkel, E. An iterative method for the Helmholtz equation. *J. Comput. Phys.* **49** (1983), no. 3, 443–457.

[2] Benamou, J.-D.; Desprès, B. A domain decomposition method for the Helmholtz equation and related optimal control problems. *J. Comput. Phys.* **136** (1997), no. 1, 68–82.

[3] Benzi, M.; Haws, J. C.; Tůma, M. Preconditioning highly indefinite and nonsymmetric matrices. *SIAM J. Sci. Comput.* **22** (2000), no. 4, 1333–1353 (electronic).

[4] Berenger, J.-P. A perfectly matched layer for the absorption of electromagnetic waves. *J. Comput. Phys.* **114** (1994), no. 2, 185–200.

[5] Biondi, B. L. *3D seismic imaging*: *three dimensional seismic imaging.* Investigations in Geophyics, 14. Society of Exploration Geomphysicists, Tulsa, Okla., 2006.

[6] Bleszynski, E.; Bleszynski, M.; Jaroszewicz, T. AIM: Adaptive integral method for solving large-scale electromagnetic scattering and radiation problems. *Radio Sci.* **31** (1996), no. 5, 1225–1252. Available at: `http://www.agu.org/journals/ABS/1996/96RS02504.shtml`

[7] Börm, S.; Grasedyck, L.; Hackbusch, W. *Hierarchical matrices*. Lecture note 21/2003. Revised 2006. Max Planck Institut für Mathematik, Bonn, Germany. Available at: `http://www.mis.mpg.de/publications/other-series/ln/lecturenote-2103.html`

[8] Brandt, A.; Livshits, I. Wave-ray multigrid method for standing wave equations. *Electron. Trans. Numer. Anal.* **6** (1997), Dec., 162–181 (electronic).

[9] Bruno, O. P.; McKay Hyde, E. Higher-order Fourier approximation in scattering by two-dimensional, inhomogeneous media. *SIAM J. Numer. Anal.* **42** (2005), no. 6, 2298–2319.

[10] Chew, W. C.; Weedon, W. H. A 3-d perfectly matched medium from modified Maxwell's equations with stretched coordinates. *Microwave Opt. Tech. Lett.* **7** (1994), 599–604. Available at: `http://www.ccem.uiuc.edu/chew/e_papers/pml3d.ps.gz`

[11] Després, B. Domain decomposition method and the Helmholtz problem. *Mathematical and numerical aspects of wave propagation phenomena (Strasbourg, 1991)*, 44–52. SIAM, Philadelphia, 1991.

[12] Duff, I. S.; Reid, J. K. The multifrontal solution of indefinite sparse symmetric linear equations. *ACM Trans. Math. Software* **9** (1983), no. 3, 302–325.

[13] Elman, H. C.; Ernst, O. G.; O'Leary, D. P. A multigrid method enhanced by Krylov subspace iteration for discrete Helmholtz equations. *SIAM J. Sci. Comput.* **23** (2001), no. 4, 1291–1315 (electronic).

[14] Engquist, B.; Majda, A. Absorbing boundary conditions for the numerical simulation of waves. *Math. Comp.* **31** (1977), no. 139, 629–651.

[15] Engquist, B.; Majda, A. Radiation boundary conditions for acoustic and elastic wave calculations. *Comm. Pure Appl. Math.* **32** (1979), no. 3, 314–358.

[16] Engquist, B.; Runborg, O. Computational high frequency wave propagation. *Acta Numer.* **12** (2003), 181–266.

[17] Engquist, B.; Ying, L. Fast directional multilevel algorithms for oscillatory kernels. *SIAM J. Sci. Comput.* **29** (2007), no. 4, 1710–1737 (electronic).

[18] Engquist, B.; Ying, L. A fast directional algorithm for high frequency acoustic scattering in two dimensions. *Commun. Math. Sci.* **7** (2009), no. 2, 327–345.

[19] Engquist, B.; Ying, L. Sweeping preconditioner for the Helmholtz equation: moving perfectly matched layers. Preprint, 2010. Available at: http://www.math.utexas.edu/users/lexing/publications/index.html

[20] Erlangga, Y. A. Advances in iterative methods and preconditioners for the Helmholtz equation. *Arch. Comput. Methods Eng.* **15** (2008), no. 1, 37–66.

[21] Erlangga, Y. A.; Oosterlee, C. W.; Vuik, C. A novel multigrid based preconditioner for heterogeneous Helmholtz problems. *SIAM J. Sci. Comput.* **27** (2006), no. 4, 1471–1492 (electronic).

[22] Erlangga, Y. A.; Vuik, C.; Oosterlee, C. W. On a class of preconditioners for solving the Helmholtz equation. *Appl. Numer. Math.* **50** (2004), no. 3-4, 409–425.

[23] Fish, J.; Qu, Y. Global-basis two-level method for indefinite systems. I. Convergence studies. *Internat. J. Numer. Methods Engrg.* **49** (2000), no. 3, 439–460.

[24] Gander, M. J.; Nataf, F. An incomplete LU preconditioner for problems in acoustics. *J. Comput. Acoust.* **13** (2005), no. 3, 455–476.

[25] George, A. Nested dissection of a regular finite element mesh. *SIAM J. Numer. Anal.* **10** (1973), 345–363.

[26] Grasedyck, L.; Hackbusch, W. Construction and arithmetics of $\mathcal{H}$-matrices. *Computing* **70** (2003), no. 4, 295–334.

[27] Hackbusch, W. A sparse matrix arithmetic based on $\mathcal{H}$-matrices. I. Introduction to $\mathcal{H}$-matrices. *Computing* **62** (1999), no. 2, 89–108.

[28] Halko, N.; Martinsson, P.-G.; Tropp, J. A. Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions. Preprint, 2009. arXiv:0909.4061v1.

[29] Johnson, S. G. Notes on perfectly matched layers (PMLs). Massachusetts Institute of Technology, Technical Report, 2007; updated 2010. Available at: http://www-math.mit.edu/~stevenj/18.369/pml.pdf

[30] Laird, A. L.; Giles, M. B. Preconditioned iterative solution of the 2D Helmholtz equation. Preprint, 2002. Technical Report, NA-02-12, Computing Laboratory, University of Oxford. Available at: ftp://ftp.comlab.ox.ac.uk/pub/Documents/techreports/NA-02-12.ps

[31] Lee, B.; Manteuffel, T. A.; McCormick, S. F.; Ruge, J. First-order system least-squares for the Helmholtz equation. *SIAM J. Sci. Comput.* **21** (2000), no. 5, 1927–1949 (electronic).

[32] Liberty, E.; Woolfe, F.; Martinsson, P.-G.; Rokhlin, V.; Tygert, M. Randomized algorithms for the low-rank approximation of matrices. *Proc. Natl. Acad. Sci. USA* **104** (2007), no. 51, 20167–20172.

[33] Liu, J. W. H. The multifrontal method for sparse matrix solution: theory and practice. *SIAM Rev.* **34** (1992), no. 1, 82–109.

[34] Livshits, I.; Brandt, A. Accuracy properties of the wave-ray multigrid algorithm for Helmholtz equations. *SIAM J. Sci. Comput.* **28** (2006), no. 4, 1228–1251 (electronic).

[35] Margrave, G. F.; Ferguson, R. J. Wavefield extrapolation by nonstationary phase shift. *Geophys.* **64** (1999), no. 4, 1067–1078. Available at: http://www.seg.org/publications/geoarchive/1999/jul-aug/geo-6404r1067.pdf

[36] Martinsson, P.-G. A fast direct solver for a class of elliptic partial differential equations. *J. Sci. Comput.* **38** (2009), no. 3, 316–330.

[37] Martinsson, P.-G.; Rokhlin, V. A fast direct solver for scattering problems involving elongated structures. *J. Comput. Phys.* **221** (2007), no. 1, 288–302.

[38] Osei-Kuffuor, D.; Saad, Y. Preconditioning Helmholtz linear systems. *Appl. Numer. Math.* **60** (2010), no. 4, 420–431.

[39] Rokhlin, V. Rapid solution of integral equations of scattering theory in two dimensions. *J. Comput. Phys.* **86** (1990), no. 2, 414–439.

[40] Rokhlin, V. Diagonal forms of translation operators for the Helmholtz equation in three dimensions. *Appl. Comput. Harmon. Anal.* **1** (1993), no. 1, 82–93.

[41] Saad, Y. *Iterative methods for sparse linear systems*. 2nd ed. Society for Industrial and Applied Mathematics, Philadelphia, 2003.

[42] Saad, Y.; Schultz, M. H. GMRES: a generalized minimal residual algorithm for solving non-symmetric linear systems. *SIAM J. Sci. Statist. Comput.* **7** (1986), no. 3, 856–869.

[43] Sandberg, K.; Beylkin, G. Full-wave-equation depth extrapolation for migration. *Geophys.* **74** (2009), no. 6, WCA121–WCA128. Available at: `http://link.aip.org/link/?GPY/74/WCA121/1`

[44] Susan-Resiga, R. F.; Atassi, H. M. A domain decomposition method for the exterior Helmholtz problem. *J. Comput. Phys.* **147** (1998), no. 2, 388–401.

[45] Vaněk, P.; Mandel, J.; Brezina, M. Two-level algebraic multigrid for the Helmholtz problem. *Domain decomposition methods, 10 (Boulder, CO, 1997)*, 349–356. Contemporary Mathematics, 218. American Mathematical Society, Providence, R.I., 1998.

[46] Xia, J.; Chandrasekaran, S.; Gu, M.; Li, X. S. Superfast multifrontal method for large structured linear systems of equations. *SIAM J. Matrix Anal. Appl.* **31** (2009), no. 3, 1382–1411.

BJÖRN ENGQUIST
University of Texas at Austin
Department of Mathematics and ICES
1 University Station, C1200
Austin, TX 78712
E-mail: `engquist@math.utexas.edu`

LEXING YING
University of Texas at Austin
Department of Mathematics and ICES
1 University Station, C1200
Austin, TX 78712
E-mail: `lexing@math.utexas.edu`

Received August 2010.