NESTED DOMAIN DECOMPOSITION WITH POLARIZED TRACES FOR THE 2D HELMHOLTZ EQUATION

LEONARDO ZEPEDA-NÚÑEZ[†][*] AND LAURENT DEMANET[†]

Abstract. We present a solver for the 2D high-frequency Helmholtz equation in heterogeneous, constant density, acoustic media, with online parallel complexity that scales empirically as $\mathcal{O}(\frac{N}{P})$, where N is the number of volume unknowns, and P is the number of processors, as long as $P = \mathcal{O}(N^{1/5})$. This sublinear scaling is achieved by domain decomposition, not distributed linear algebra, and improves on the $P = \mathcal{O}(N^{1/8})$ scaling reported earlier in [103]. The solver relies on a two-level nested domain decomposition: a layered partition on the outer level, and a further decomposition of each layer in cells at the inner level. The Helmholtz equation is reduced to a surface integral equation (SIE) posed at the interfaces between layers, efficiently solved via a nested version of the polarized traces preconditioner [103]. The favorable complexity is achieved via an efficient application of the integral operators involved in the SIE.

1. Introduction. It has become clear over the past several years that the right mix of ideas to obtain an efficient Helmholtz solver, in the high-frequency regime, involves domain decomposition with accurate transmission conditions. The first empirical $\mathcal{O}(N)$ complexity algorithm, where N is the number of degrees of freedom, was the sweeping preconditioner of Engquist and Ying [33] that uses a decomposition in grid-spacing-thin layers coupled with an efficient multi-frontal solver at each layer. Subsequently, Stolk proposed different instances of domain decomposition methods [88, 89] that restored the ability to use arbitrarily thick layers, improving the efficiency of the local solves at each layer, with the same $\mathcal{O}(N)$ claim. Liu and Ying recently presented a recursive version of the sweeping preconditioner in 3D that decreases the offline cost to linear complexity [68] and a variant of the sweeping preconditioner using an additive Schwarz preconditioner [67]. Many other authors have proposed algorithms with similar complexity claims that we review in Section 1.2, which we briefly compare in Section 1.3.

Although these algorithms are instances of efficient iterative methods, they have to revisit all the degrees of freedom inside the volume in a sequential (or sweeping) fashion at each iteration. This sequential computation thus hinders the scalability of the algorithms to large-scale parallel architectures.

Two solutions have been proposed so far to mitigate the lack of asymptotic scalability in the high-frequency regime:

- Poulson et al. [80] parallelized the sweeping preconditioner in 3D by using distributed linear algebra to solve the local system at each layer, obtaining a $\mathcal{O}(N)$ complexity. This approach should in principle reach sublinear complexity scalings in 2D. The resulting codes are complex, and using distributed linear algebra libraries can be cumbersome for industrial application because of licensing issues.
- Zepeda-Núñez and Demanet [103] proposed the method of polarized traces, with an online empirical runtime $\mathcal{O}(N/P)$ in 2D, where N is the number of degrees of freedom in the volume[1], and P is the number of nodes, in a distributed memory environment, provided that $P = \mathcal{O}(N^{1/8})$, and that the

[1]Throughout this paper, the notation $\mathcal{O}(\cdot)$ may hide logarithmic factors.

medium does not contain micro-structures at the wavelength level with a preferred orientation aligned with the sweeping direction[2]. The communication cost is a negligible $\mathcal{O}(N^{1/2}P)$.

In this paper we follow, and improve on the latter approach. The method of polarized traces relies on a layered domain decomposition coupled with an surface integral equation (SIE) posed at the interfaces between subdomains that is easy to precondition. The algorithm has two stages: an expensive but parallel offline stage that is performed only once, and a fast online stage that is performed for each right-hand side (source). The above-mentioned online complexity is the result of: first, the efficient preconditioner; and second, the precomputation and compression of the operators involved in the SIE and its preconditioner during the offline stage.

The main improvements of the solver presented in this paper stem from a nested layered domain decomposition in two levels. In the outer level we use an equivalent matrix-free formulation of the SIE that relies on solving a local problem within each layer, with equivalent sources at the interfaces. In the inner level, we decompose each layer in cells and we use the original method of polarized traces to solve the local problems efficiently, via a local SIE posed at the interfaces between cells within a layer. As it will be explained in the sequel, the operators involved in the local SIE at each local problem are much smaller, making them cheaper to precompute and to apply. Finally, the number of layers and cells can be balanced to increase the parallelism and hence reduce the asymptotic runtime.

1.1. Results. We propose a variant of the method of polarized traces using a nested domain decomposition approach. This novel approach results in an algorithm with an empirical asymptotic online runtime of $\mathcal{O}(N/P)$ provided that $P = \mathcal{O}(N^{1/5})$, which results in a lower asymptotic online runtime in a distributed memory environment. Moreover, the nested polarized traces method also has lower memory footprint[3], and lower offline complexity, as shown in Table 1.1. The parameter α in Table 1.1 is the exponent of the empirical asymptotic complexity[4].

The nested polarized traces method inherits the modularity from the original method of polarized traces; it can easily be extended to more complex physics and higher order discretizations; and it takes advantage of new developments in direct methods such as [52, 83, 99] and in better block-low-rank and butterfly matrix compression techniques such as [6, 12, 65].

In addition, we propose a few minor improvements to the original scheme presented in [103] to obtain better accuracy, and to accelerate the convergence rate. We provide :

1. an equivalent formulation of the method of polarized traces that involves a discretization using Q1 finite elements on a Cartesian grid using a suitable quadrature rule to compute the mass matrix. This formulation can be easily generalized to high-order finite differences and high-order finite elements, thus circumventing the labor-intensive summation by parts;

---

[2] As it will be explained in the sequel, the combination of higher frequencies and thinner subdomains increases the number of iterations, thus resulting in a mild deterioration of the scalings.

[3] The offline complexity for the method of polarized in this case (Table 1.1) is higher than in [103] because we assume that we have P nodes instead of $N^{1/2}P$.

[4] We choose to express the complexity with respect to the global number of degrees of freedom, N, in order to provide a cleaner expression in Table 1.1. of the application of a compressed numerical Green's function with respect to N. The exact definition of α is given in Section 4.3. In this case, the value of $\alpha = 3/4$ is estimated empirically

Stage	Polarized traces	Nested polarized traces
offline	$\mathcal{O}\left(N^{3/2}/P\right)$	$\mathcal{O}\left((N/P)^{3/2}\right)$
online	$\mathcal{O}\left(PN^{\alpha} + N/P\right)$	$\mathcal{O}\left(P^{1-\alpha}N^{\alpha} + N/P\right)$

<div align="center">TABLE 1.1</div>

Runtime of both formulations (up to poly-logarithmic factors), supposing only P nodes, one node per layer for the method of polarized traces, and one node per cell for the nested domain decomposition with polarized traces. The value of α depends on the ability to compress local Green's functions in partitioned low-rank form; there exist theoretical and practical arguments for setting $\alpha = 3/4$. In that case, the N/P contribution dominates $P^{1-\alpha}N^{\alpha}$ as long as $P = O(N^{1/5})$. These scalings hold provided that the medium does not contain micro-structures at the wavelength level presenting a preferred orintation aligned with the sweeping direction.

2. and, a variant of the preconditioner introduced in [103], in which we used a block Gauss-Seidel iteration instead of a block Jacobi iteration, that improves the convergence rate.

Even though the nested approach presented in this paper has a lower asymptotic online runtime than the method of polarized traces, in practice, the latter is faster. This is due to the large constants resulting from iterating within each layer in the nested approach. In order to reduce the constants, we introduce a small variant of the nested approach, which relies on a compressed LU factorization to solve the local SIE within each layer. The asymptotic online runtimes remains unchanged, with much lower constants albeit with a more thorough precomputation.

1.2. Related work. Using domain decomposition to solve PDE's can be dated back to Schwarz [86] and Lions [66], in which the Laplace equation is solved iteratively. However, using such techniques to solve the Helmholtz problem was proposed for the first time by Després in [29], which led to the development of many different approaches focusing mostly on the discretization of the Helmholtz equation; in particular: the ultra weak variational formulation (UWVF) [17], which, in return, spawned plane wave methods such as the Trefftz formulation of Perugia et al. [76], the plane wave discontinuous Galerkin method [53, 58], the discontinuous enrichment method of Farhat et al. [39], the partition of unity method (PUM) by Babuska and Melenk [4], the least-squares method by Monk and Wang [78], and, more recently, the multi-trace formulation of Hiptmair and Jerez-Hanckes [24, 57], among many others. A recent and thorough review can be found in [59].

Moreover, the ideas of Lions and Després lead to the development of various domain decomposition algorithms, which can be classified as Schwarz algorithms (for a review on classical Schwarz methods see [18, 90]) with or without overlap for the Helmholtz equation [13, 25, 28, 51, 70, 74]. However, soon it became evident that the convergence rate of such algorithms was spectacularly dependent on the boundary conditions prescribed at the interfaces between subdomains.

In the quest to design boundary conditions that ensured a fast convergence Gander introduced the framework of optimized Schwarz methods in [40]. Within that framework, Gander et al. provided an optimal non-local boundary condition, which is then approximated by an optimized Robin boundary condition [43]. How to design better approximations has been studied in [42, 45, 46, 48]. More recently Boubendir et al. [14] presented a quasi-optimal optimized Schwarz method using Padé approximations of the Dirichlet to Neumann (DtN) map.

The idea of mixing domain decomposition and absorbing boundary condition was first explored by Engquist and Zhao [34] for elliptic problems. The application of such

ideas to the Helmholtz problem can be traced back, to great extent, to the AILU pre-conditioner of Gander and Nataf [44], in which a layered domain decomposition was used; and to Plessix and Mulder [79] in which a similar idea is used using separation of variables. However, it was Engquist and Ying who showed in [33, 32] that such ideas could yield fast methods to solve the high-frequency Helmholtz equation, by introducing the sweeping preconditioner, which was then extended by Tsuji and collaborators to different discretizations and physics [92, 93, 94]. Since then, many other papers have proposed methods with similar claims. Stolk [88] proposed a domain decomposition method using single layer potentials to transfer the information between subdomains. Geuzaine and Vion explored randomized techniques [23] to probe the DtN map [8] in order to approximate the absorbing boundary conditions within a multiplicative Schwarz iteration [96, 97]. Chen and Xiang proposed another instance of efficient domain decomposition where the emphasis is on transferring sources from one subdomain to another [21, 22], which has inspired similar methods [64, 63, 30]. Most of the methods mentioned above can be recast as optimized Schwarz methods (cf. [20]). Closely related to the content of this paper, we find the method of polarized traces [103], and an earlier version of this work [102].

Luo et al. proposed the fast sweeping Huygens method, based on an approximate Green's function computed via geometric optics, coupled with a butterfly algorithm, which can handle transmitted waves in very favorable complexity [69, 81].

Alongside iterative methods, some advances have also been made on multigrid methods. Brandt and Livshits developed the wave-ray method [15], in which the oscillation error is eliminated by a ray-cycle using a geometric optics approximation; Haber and MacLachlan proposed an alternative formulation that can be solved by standard multigrid methods [56]; Erlangga et al. [37] showed how to implement a simple, although suboptimal, complex-shifted Laplace preconditioner with multigrid. Another variant of complex-shifted Laplacian method with deflation was studied by Sheikh et al. [87]. The choice of the optimal complex-shift was studied by Cools and Vanroose [26] and by Gander et al. [49]. Finally, the generalization of the complex shifted Laplacian preconditioner to the elastic wave equation has been studied by Rizzuti and Mulder [82]. We point out that most of the multigrid methods mentioned above exhibit a suboptimal dependence of the number of iterations to converge with respect to the frequency, making them ill-suited for high frequency problems. However, they are easy to parallelize resulting in small runtimes, as shown by Calandra et al. [16].

A good early review of iterative methods for the Helmholtz equation is in [36]. Another review paper that discussed the difficulties generated by the high-frequency limit is [38].

In another exciting direction, much progress has been made on making direct methods efficient for the Helmholtz equation. Such is the case of Wang et al.'s method [99], which couples nested dissection and multi-frontal elimination with $\mathcal{H}$-matrices (see [31] for the multi-frontal method, and [50] for nested dissection). Another example is the work of Gillman, Barnett and Martinsson on computing Dirichlet to Neumann maps in a multiscale fashion [52]. Recently, Ambikasaran et al. [1] proposed a direct solver for acoustic scattering for heterogeneous media using compressed linear algebra to solve an equivalent Lippmann-Schwinger equation. These methods are extremely efficient for elliptic and low frequency problems due to the high compressibility of the Green's functions [7]. However, it is not yet clear whether offline linear complexity scalings for high-frequency problems can be achieved this way (cf. [35]), though good

direct methods are often faster in practice than the iterative methods mentioned at the beginning of this Section. The main issue with direct methods is the lack of scalability to very large-scale problems due to the memory requirements and prohibitively expensive communication overheads.

Finally, beautiful mathematical reviews of the Helmholtz equation are [77] and [19]. A more systematic and extended exposition of the references mentioned here can be found in [101].

1.3. Comparison to other preconditioners. There has been a recent effort to express the fast preconditioners based on domain decomposition and absorbing boundary conditions in terms of optimized Schwarz method [20, 47, 55], in particular, the method of polarized traces can be re-casted, algorithmically, as an optimized Schwarz method; however, the polarization of the waves using discrete integral relations, and in particular, the recasting of the volumetric problem as an extended algebraically exact integral problem posed at at the interfaces between layers, seems to escape the general framework. We do not aim to compare the method of polarized traces nor its nested variant under the framework of optimized Schwarz methods, which would be out of the scope of this paper. Instead, we aim to briefly compare the method of polarized traces against some related methods in the literature[5] .

We start by comparing the method of polarized traces to the original sweeping preconditioner. The sweeping preconditioner is based on a block LDL^t factorization, coupled with the remarkable observation that the inverse of the blocks of D correspond to a half-space problem. The inverse of the blocks can be either compressed using $\mathcal{H}$-matrices[6] [32] or applied using an auxiliary problem [33], in which the half-space problem is truncated using a high-quality PML. The sweeping preconditioner acts on the volume problem, and it needs to revisit all the degrees of freedom, thus hindering parallelization, which is mostly achieved by parallelized linear algebra [80]. On the other hand, the method of polarized traces preconditions a SIE posed on the interfaces reducing the number of degrees of freedom. The application of the SIE and the preconditioner can be parallelized and accelerated using fast summation algorithms, and the number of iterations to convergence is, in general, much smaller.

Another method based on the sweeping preconditioner is the source transfer method [21], an overlapping domain decomposition. In this method the residual in a generous overlap, times a smooth window, is used to propagate the information to the neighboring domains, in a fashion that resembles an iterative refinement iteration. If the window is chosen as a Heaviside function; the source transfer method has some striking similarities to the rapidly converging DDM method [88] and the method of polarized traces. However, the source transfer method is a preconditioner for the volume problem, needing to revisit all the degrees of freedom each iteration, and the generous overlap between subdomain can be prohibitively expensive in 3D.

The closest algorithm to the method of polarized traces is the rapidly converging domain decomposition method of Stolk [88], in which the information is transfered between subdomains using a single layer potential in [88] and using both single and double layer potential in [89]. The later has the same form as the sweeps in the method of polarized traces (see [104]). Stolk's method preconditions the volume problem, which requires him to move the boundaries of the subdomains and to use

---

[5]We refer the interested reader to [41] for a short exposition on possible shortcomings of sweeping-like preconditioners.

[6]This is possible, in homogeneous media, thanks to the compressibility of the Green's function when considering elongated structured [72].

a step of iterative refinement between sweeps. In practice, Stolk's method has lower constants but a higher asymptotic runtime in 2D.

Finally, we compare the method of polarized traces to the double sweep preconditioner, which is, by construction, an optimized Schwarz method. In [97] the Helmholtz problem is posed as a boundary problem between interfaces using different boundary conditions. The boundary problem is then solved iteratively, using an Schwarz iteration accelerated with preconditioned GMRES. The main disadvantage of the double sweep preconditioner is its complexity. To converge in $\mathcal{O}(1)$ iterations, the double sweep preconditioner requires a good knowledge of the DtN map, which is probed using randomized methods (cf. [8]) during an offline stage. As a consequence, the linear systems at each subdomain have a non-local boundary condition, which results in matrices with large dense blocks, making the local linear systems expensive to solve.

At the level of the formulation there exists some differences between the two methods; in [97] the resulting system posed on the the interfaces has a block-tridiagonal structure, which is hard to solve iteratively; however, some off-diagonal blocks, which represent self-interactions within layers, are small, so they can be neglected. The resulting sparsified system has an interlaced structure and it can be solved by back-substitution, in which each block is a local solve with a non-local boundary condition. In contrast, the method of polarized traces transitions from a volumetric discretization to an extended *equivalent* boundary problem, without approximation making it better suited to cases in which the self-interactions, such as multiple scattering within a layer, can not be neglected. By exploiting the block structure of the latter we can easily construct an efficient preconditioner based on a block Gauss-Seidel iteration within a GMRES [85] or BiCGstab [95] iteration.Finally, we point out that the different types of sweeps described in [98] can be seamlessly deduced using different block splittings of the extended SIE.

1.4. Organization. The present paper is organized as follows :
- we review the formulation of the Helmholtz problem and the reduction to a boundary integral equation in Section 2;
- in Section 3 we review the method of polarized traces;
- in Section 4 we present two variants of the nested solver, and we provide the empirical complexity observed;
- finally, in Section 5 we present numerical experiments that corroborate the complexity claims.

2. Formulation. Let Ω be a rectangle in $\mathbb{R}^2$, and consider a layered partition of Ω into L slabs, or layers $\{\Omega^\ell\}_{\ell=1}^L$ as shown in Fig 2.1. Define the squared slowness as $m(\mathbf{x}) = 1/c(\mathbf{x})^2$, $\mathbf{x} = (x, z)$. As in geophysics, we may refer to z as depth, and we suppose that it points downwards. Define the global Helmholtz operator at frequency ω as

$$\mathcal{H}u = \left(-\triangle - m\omega^2\right) u \qquad \text{in } \Omega, \tag{2.1}$$

with an absorbing boundary condition on $\partial\Omega$ implemented via perfectly matched layers (PML) [9, 10, 61].

Let us define f^ℓ as the restriction of f to Ω^ℓ, i.e., $f^\ell = f\chi_{\Omega^\ell}$, where χ_{Ω^ℓ} is the characteristic function of Ω^ℓ. Define the local Helmholtz operators as

$$\mathcal{H}^\ell u = \left(-\triangle - m\omega^2\right) u \qquad \text{in } \Omega^\ell, \tag{2.2}$$

with absorbing boundary conditions on $\partial\Omega^\ell$. Finally, let u be the solution to $\mathcal{H}u = f$.

As for any domain decomposition method we seek to find u by solving the local systems $\mathcal{H}^\ell v^\ell = f^\ell \chi_{\Omega^\ell}$. In order to compute u, classical domain decomposition methods (cf. [90]) requires a coupling between the subdomains, which usually takes the form of continuity or boundary conditions. In this case, the global coupling between subdomains is realized via a reduction of the problem posed on the volume to a problem posed on the interfaces between layers, resulting in a SIE. The main tool used in this endeavor is the Green's representation formula (GRF) in each layer.

If the global Green's function G, given by $\mathcal{H}G(\mathbf{x}, \mathbf{y}) = \delta(\mathbf{x} - \mathbf{y})$, is known, then we can write in the interior of each layer

$$u(\mathbf{x}) = Gf^\ell(\mathbf{x}) + \int_{\partial\Omega^\ell} \left(G(\mathbf{x}, \mathbf{y})\partial_{\nu_y} u(\mathbf{y})dy - \partial_{\nu_y} G(\mathbf{x}, \mathbf{y})u(\mathbf{y}) \right) dS_{\mathbf{y}}, \qquad (2.3)$$

for $\mathbf{x} \in \Omega^\ell$, and $Gf^\ell(\mathbf{x}) = \int_\Omega G(\mathbf{x}, \mathbf{y})f^\ell(\mathbf{y})d\mathbf{y}$. One remarkable, and mostly undervalued, property of the GRF is that (2.3) remains true even if we change the Green's function for a *local* Green's function, which we denote G^ℓ, provided that

$$\mathcal{H}G^\ell(\mathbf{x}, \mathbf{y}) = \delta(\mathbf{x} - \mathbf{y}) \qquad (2.4)$$

is satisfied for $\mathbf{x}, \mathbf{y} \in \Omega^\ell$, where $\mathcal{H}$ acts on $\mathbf{x}$.

Using the local GRF, the solution can be written *without approximation* in each layer as

$$u(\mathbf{x}) = G^\ell f^\ell(\mathbf{x}) + \int_{\partial\Omega^\ell} \left(G^\ell(\mathbf{x}, \mathbf{y})\partial_{\nu_y} u(\mathbf{y})dy - \partial_{\nu_y} G^\ell(\mathbf{x}, \mathbf{y})u(\mathbf{y}) \right) dS_{\mathbf{y}} \qquad (2.5)$$

for $\mathbf{x} \in \Omega^\ell$, where $G^\ell f^\ell(\mathbf{x}) = \int_{\Omega^\ell} G^\ell(\mathbf{x}, \mathbf{y})f^\ell(\mathbf{y})d\mathbf{y}$ and $G^\ell(\mathbf{x}, \mathbf{y})$ is the solution of $\mathcal{H}^\ell G^\ell(\mathbf{x}, \mathbf{y}) = \delta(\mathbf{x} - \mathbf{y})$. Where $\mathcal{H}^\ell$ coincides with $\mathcal{H}$ in Ω^ℓ by construction, thus G^ℓ satisfies (2.4).

Denote $\Gamma_{\ell,\ell+1} = \partial\Omega^\ell \cap \partial\Omega^{\ell+1}$. Supposing that Ω^ℓ are thin slabs either extending to infinity, or surrounded by a damping layer on the lateral sides, we can rewrite (2.5) as

$$u(\mathbf{x}) = G^\ell f^\ell(\mathbf{x}) - \int_{\Gamma_{\ell-1,\ell}} G^\ell(\mathbf{x}, \mathbf{x}')\partial_z u(\mathbf{x}')dx' + \int_{\Gamma_{\ell,\ell+1}} G^\ell(\mathbf{x}, \mathbf{x}')\partial_z u(\mathbf{x}')dx'$$
$$+ \int_{\Gamma_{\ell-1,\ell}} \partial_z G^\ell(\mathbf{x}, \mathbf{x}')u(\mathbf{x}')dx' - \int_{\Gamma_{\ell,\ell+1}} \partial_z G^\ell(\mathbf{x}, \mathbf{x}')u(\mathbf{x}')dx'. \qquad (2.6)$$

The knowledge of u and $\partial_z u$ on the interfaces $\Gamma_{\ell,\ell+1}$ therefore suffices to recover the solution everywhere in Ω. We show in Section 2.2 how to build, via an algebraic reduction, a discrete SIE (based on (2.6)) posed on the interfaces between boundaries $\{\Gamma_{\ell,\ell+1}\}_{\ell=1}^{L-2}$, and whose solution is exactly the restriction of the global solution to the interfaces between layers. Once the solution at the interfaces is known, the solution can be reconstructed at each layer exactly using (2.6). The remaining question is how to efficiently solve the SIE using an iterative method. The answer lies within the concept of polarization, which is the main topic of the next section.

2.1. Polarization. In this section we provide the rationale for the polarization of waves, concept that plays a crucial role to solve the SIE mentioned above efficiently. We say that a wave is *polarized* at an interface when it is generated by sources supported only on one side of that interface.
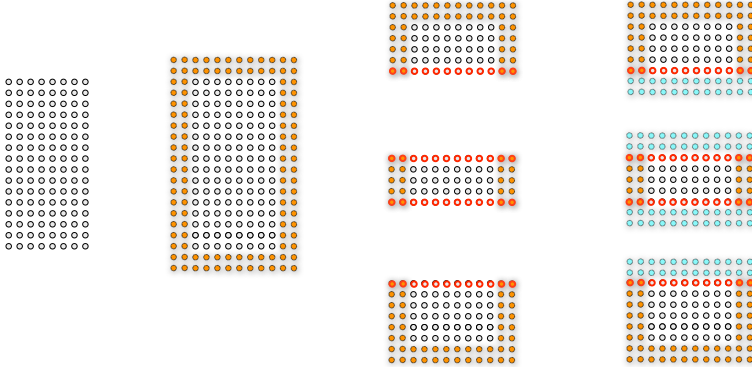
FIG. 2.1. *Layered domain decomposition. The orange grid-points represent the PML for the original problem, the light-blue represent the artificial PML between layers, and the red grid-points correspond to $\underline{\mathbf{u}}$ in* (2.11).

In order to express the polarizing conditions in boundary integral form, we briefly recall the rationale provided in [103]. Let consider Fig. 2.2, where $\mathbf{x} = (x, z)$ with z pointing down. Consider a interface Γ partitioning $\mathbb{R}^2$ as $\Omega^{\mathrm{down}} \cup \Omega^{\mathrm{up}}$, with $f \neq 0$ in Ω^{down}. Let $\mathbf{x} \in \Omega^{\mathrm{up}}$, and consider a contour made up of Γ and a semi-circle D at infinity in the upper half-plane Ω^{up}. In case the wave speed becomes uniform past some large radius, the Sommerfeld radiation condition (SRC) puts to zero the contribution on D in Green's representation formula (GRF) [62, 75] resulting in the *incomplete Green's formula*

$$u(\mathbf{x}) = \int_{\Gamma} \left(\frac{\partial G}{\partial z_y}(\mathbf{x}, \mathbf{y}) u(\mathbf{y}) - G(\mathbf{x}, \mathbf{y}) \frac{\partial u}{\partial z_y}(\mathbf{y}) \right) dS_{\mathbf{y}}, \qquad \mathbf{x} \in \Omega_{\mathrm{up}} \setminus \Gamma. \qquad (2.7)$$

On the other hand, if x approaches Γ from below, then we obtain the *annihilation formula*

$$0 = \int_{\Gamma} \left(\frac{\partial G}{\partial z_y}(\mathbf{x}, \mathbf{y}) u(\mathbf{y}) - G(\mathbf{x}, \mathbf{y}) \frac{\partial u}{\partial z_y}(\mathbf{y}) \right) dS_{\mathbf{y}}, \qquad \mathbf{x} \to \Gamma \text{ from below.} \qquad (2.8)$$

We can observe that (2.7) and (2.8) are equivalent; either one can be used as the definition of a polarizing boundary condition on Γ. In this case we can say that a wavefield is up-going with respect to Γ if it was generated by a source located below γ, or equivalently, if its Dirichlet and Neumann traces, denoted $u^{\uparrow}$ and $\partial_z u^{\uparrow}$, satisfy the annihilation condition (2.8). We can easily express the same annihilation condition for a down-going wavefield by the annihilation formula

$$0 = \int_{\Gamma} \left(-\frac{\partial G}{\partial z_y}(\mathbf{x}, \mathbf{y}) u(\mathbf{y}) + G(\mathbf{x}, \mathbf{y}) \frac{\partial u}{\partial z_y}(\mathbf{y}) \right) dS_{\mathbf{y}}, \qquad \mathbf{x} \to \Gamma \text{ from above.} \qquad (2.9)$$

In order to efficiently solve the globally coupling SIE, we define an extended system by introducing extra variables. We split $u = u^{\uparrow} + u^{\downarrow}$ and $\partial_z u = \partial_z u^{\uparrow} + \partial_z u^{\downarrow}$ on $\Gamma_{\ell, \ell+1}$, by letting $(u^{\uparrow}, \partial_z u^{\uparrow})$ be polarized up in Ω^{ℓ} (according to (2.8)), and $(u^{\downarrow}, \partial_z u^{\downarrow})$ polarized down (according with (2.9)) in $\Omega^{\ell+1}$. Together, the interface fields $u^{\uparrow}, u^{\downarrow}, \partial_z u^{\uparrow}, \partial_z u^{\downarrow}$ are the polarized traces that serve as unknowns for the numerical method.
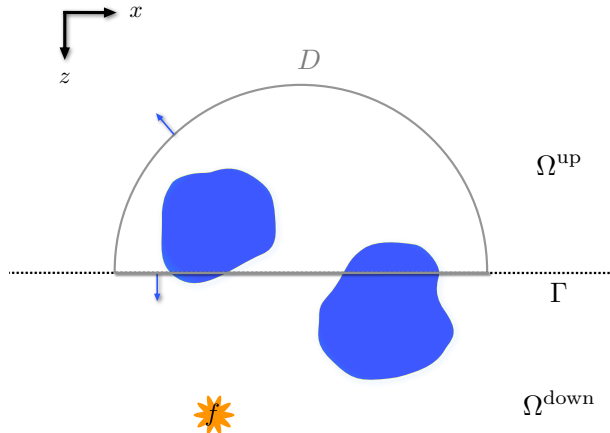
FIG. 2.2. *Illustration of* (2.7) *and* (2.8).

The discrete system is then set up from discrete algebraic reformulations[7] of the local GRF (2.6) with the polarizing conditions (2.8) and (2.9), in a manner that will be made explicit in Section 3. The resulting system has a 2-by-2 structure with block-triangular submatrices on the diagonal, and comparably small off-diagonal submatrices. A very good preconditioner consists in inverting the block-triangular submatrices by back- and forward-substitution. One application of this preconditioner can be seen as a sweep of the domain to compute transmitted (as well as locally reflected) waves using (2.7). From extensive numerical experiments provided in [103], the structure of the system ensures that the number of GMRES iterations grows at most as $\log \omega$, provided that the m does not contain a large resonant cavity nor micro-structures at the wavelength level with a preferred orientation aligned with the domain decomposition.

The rationale presented above can be seamlessly translated to the algebraic level, as it will be done in Section 3, in which, for the sake of reproducibility, we follow a mostly algebraic description of the algorithm.

2.2. Discrete Realization. In order to compress the notation, and yet provide enough details so that the reader can implement the algorithm presented in this paper, we recall the notation used in [103].

We can discretize (2.1) using second order finite differences (Appendix A) or Q1 finite elements (Appendix B), and we obtain the global linear system

$$\mathbf{H} \mathbf{u} = \mathbf{f}, \tag{2.10}$$

which we aim to solve using a domain decomposition approach that relies on solving the local systems $\mathbf{H}^\ell$, which are the discrete version of (2.2).

As it was explained in the prequel, the coupling between the subdomains is realized via an *equivalent* discrete SIE, which relies on a discrete version of (2.6). In this section we briefly explain, at an algebraic level, the reduction of (2.10) to an equivalent discrete SIE of the form

$$\underline{\mathbf{M}} \underline{\mathbf{u}} = \underline{\mathbf{f}}, \tag{2.11}$$

---

[7]The algebraic reformulation is performed using either summation by parts, as done in [103], or the much less labor intensive approach shown in Appendix C.

where $\underline{\mathbf{u}}$ are the degrees of freedom of $\mathbf{u}$ at the interfaces between layers (see Fig. 2.1) and $\underline{\mathbf{M}}$ is defined below after some basic notation has been introduced.

We suppose that the full domain has $N = n_x \times n_z$ discretization points, that each layer has $n_x \times n^\ell$ discretization points, and that the number of points in the PML, n_{pml}, is the same in both dimension and in every subdomain (light blue and orange nodes in Fig. 2.1).

In both discretizations the mesh is structured so that we can define $\mathbf{x}_{p,q} = (x_p, z_q) = (ph, qh)$. We assume the same ordering as in [103], i.e.

$$\mathbf{u} = (\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_{n_z}), \tag{2.12}$$

and we use the notation

$$\mathbf{u}_j = (u_{1,j}, u_{2,j}, ..., u_{n_x,j}), \tag{2.13}$$

for the entries of $\mathbf{u}$ sampled at constant depth z_j. We write $\mathbf{u}^\ell$ for the wavefield defined locally at the ℓ-th layer, i.e., $\mathbf{u}^\ell = \chi_{\Omega^\ell}\mathbf{u}$, and $\mathbf{u}_k^\ell$ for the values at the local depth[8] z_k^ℓ of $\mathbf{u}^\ell$. In particular, $\mathbf{u}_1^\ell$ and $\mathbf{u}_{n^\ell}^\ell$ are the top and bottom rows[9] of $\mathbf{u}^\ell$ (see red grid-points in Fig. 4.2). We then gather the interface traces in the vector

$$\underline{\mathbf{u}} = \left(\mathbf{u}_{n^1}^1, \mathbf{u}_1^2, \mathbf{u}_{n^2}^2, ..., \mathbf{u}_1^{L-1}, \mathbf{u}_{n^{L-1}}^{L-1}, \mathbf{u}_1^L\right)^t. \tag{2.14}$$

Define the numerical local Green's function in layer ℓ by

$$\mathbf{H}^\ell \mathbf{G}^\ell(\mathbf{x}_{i,j}, \mathbf{x}_{i',j'}) = \mathbf{H}^\ell \mathbf{G}_{i,j,i',j'}^\ell = \delta(\mathbf{x}_{i,j} - \mathbf{x}_{i',j'}), \tag{2.15}$$

if $(i,j) \in [\![-n_{\mathrm{pml}} + 1, n_x + n_{\mathrm{pml}}]\!] \times [\![-n_{\mathrm{pml}} + 1, n^\ell + n_{\mathrm{pml}}]\!]$;[10] where the discrete dirac delta[11] is given by

$$\delta(\mathbf{x}_{i,j} - \mathbf{x}_{i',j'}) = \begin{cases} \frac{1}{h^2}, & \text{if } \mathbf{x}_{i,j} = \mathbf{x}_{i',j'}, \\ 0, & \text{if } \mathbf{x}_{i,j} \neq \mathbf{x}_{i',j'}, \end{cases} \tag{2.16}$$

and where the operator $\mathbf{H}^\ell$ acts on the (i,j) indices.

Furthermore, for notational convenience we consider $\mathbf{G}^\ell$ as an operator acting on unknowns at the interfaces, as follows.

DEFINITION 2.1. *We consider* $\mathbf{G}^\ell(z_j, z_k)$ *as the linear operator defined from* $[\![-n_{pml} + 1, n_x + n_{pml}]\!] \times \{z_k\}$ *to* $[\![-n_{pml} + 1, n_x + n_{pml}]\!] \times \{z_j\}$ *given by*

$$\left(\mathbf{G}^\ell(z_j, z_k)\mathbf{v}\right)_i = h \sum_{i'=-n_{pml}+1}^{n_x+n_{pml}} \mathbf{G}^\ell((x_i, z_j), (x_{i'}, z_k))\mathbf{v}_{i'}, \tag{2.17}$$

where $\mathbf{v}$ *is a vector in* $\mathbb{C}^{n_x+2n_{pml}}$, *and* $\mathbf{G}^\ell(z_j, z_k)$ *are matrices in* $\mathbb{C}^{(n_x+2n_{pml})\times(n_x+2n_{pml})}$. *Moreover,* (2.17) *is the discrete counterpart of*

$$\int_{\mathbb{R}} G^\ell((x, z_j), (x', z_k))v(x', z_k)dx'. \tag{2.18}$$

---

[8] We hope that there is little risk of confusion in overloading z_j (local indexing) for $z_{n_c^\ell+j}$ (global indexing), where $n_c^\ell = \sum_{j=1}^{\ell-1} n^j$ is the cumulative number of points in depth.

[9] We do not consider the PML points here.

[10] The set denoted by $[\![a, b]\!]$ is equivalent to $\{i \in \mathbb{Z}; \ a \le i \le b\}$

[11] The definition is for a finite difference discretization. We refer the reader to the Appendix C for a more general definition.

10

The interface-to-interface operator $\mathbf{G}^\ell(z_j, z_k)$ is indexed by two depths – following the jargon from fast methods we call them source depth (z_k) and target depth (z_j). In particular, it represents the wavefield sampled at z_j produced by a source (in this case a measure) located at z_k.

DEFINITION 2.2. *We consider $\mathcal{G}_j^{\uparrow,\ell}(\mathbf{v}_{n^\ell}, \mathbf{v}_{n^\ell+1})$, the up-going local incomplete Green's integral; and $\mathcal{G}_j^{\downarrow,\ell}(\mathbf{v}_0, \mathbf{v}_1)$, the down-going local incomplete Green's integral, as defined by:*

$$\mathcal{G}_j^{\uparrow,\ell}(\mathbf{v}_{n^\ell}, \mathbf{v}_{n^\ell+1}) = \mathbf{G}^\ell(z_j, z_{n^\ell+1}) \left(\frac{\mathbf{v}_{n^\ell+1} - \mathbf{v}_{n^\ell}}{h} \right) \tag{2.19}$$

$$- \left(\frac{\mathbf{G}^\ell(z_j, z_{n^\ell+1}) - \mathbf{G}^\ell(z_j, z_{n^\ell})}{h} \right) \mathbf{v}_{n^\ell+1},$$

$$\mathcal{G}_j^{\downarrow,\ell}(\mathbf{v}_0, \mathbf{v}_1) = -\mathbf{G}^\ell(z_j, z_0) \left(\frac{\mathbf{v}_1 - \mathbf{v}_0}{h} \right) + \left(\frac{\mathbf{G}^\ell(z_j, z_1) - \mathbf{G}^\ell(z_j, z_0)}{h} \right) \mathbf{v}_0, \tag{2.20}$$

In the sequel we use the shorthand notation $\mathbf{G}^\ell(z_j, z_k) = \mathbf{G}_{j,k}^\ell$ when explicitly building the matrix form of the integral systems.

We can observe that Def. 2.2 is the discrete counterpart of (2.7), in which we used two neighboring traces to define the normal derivative. The expression above is the result of a discrete Green's representation formula that can be deduced from a laborious summation by parts (for more details, see [103]).

Finally, we define the Newton potential as resulting from a local solve inside each layer.

DEFINITION 2.3. *Consider the local Newton potential $\mathcal{N}_k^\ell$ applied to a local source $\mathbf{f}^\ell$ as*

$$\mathcal{N}_k^\ell \mathbf{f}^\ell = \sum_{j=1}^{n^\ell} \mathbf{G}^\ell(z_k, z_j) \mathbf{f}_j^\ell. \tag{2.21}$$

By construction $\mathcal{N}^\ell \mathbf{f}^\ell$ satisfies the equation $\left(\mathbf{H}^\ell \mathcal{N}^\ell \mathbf{f}^\ell \right)_{i,j} = \mathbf{f}_{i,j}$ for $-n_{pml} + 1 \leq i \leq n_x + n_{pml}$ and $1 \leq j \leq n^\ell$.

Following the notation introduced above, the discrete SIE reduction of the original discrete Helmholtz equation, issued from (2.6), takes the form:

$$\mathcal{G}_1^{\downarrow,\ell}(\mathbf{u}_0^\ell, \mathbf{u}_1^\ell) + \mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^\ell, \mathbf{u}_{n^\ell+1}^\ell) + \mathcal{N}_1^\ell \mathbf{f}^\ell = \mathbf{u}_1^\ell, \tag{2.22}$$

$$\mathcal{G}_{n^\ell}^{\downarrow,\ell}(\mathbf{u}_0^\ell, \mathbf{u}_1^\ell) + \mathcal{G}_{n^\ell}^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^\ell, \mathbf{u}_{n^\ell+1}^\ell) + \mathcal{N}_{n^\ell}^\ell \mathbf{f}^\ell = \mathbf{u}_{n^\ell}^\ell, \tag{2.23}$$

$$\mathbf{u}_{n^\ell}^\ell = \mathbf{u}_0^{\ell+1}, \qquad \mathbf{u}_{n^\ell+1}^\ell = \mathbf{u}_1^{\ell+1}, \tag{2.24}$$

if $1 < \ell < L$, with

$$\mathcal{G}_{n^1}^{\uparrow,1}(\mathbf{u}_{n^1}^1, \mathbf{u}_{n^1+1}^1) + \mathcal{N}_{n^1}^1 \mathbf{f}^1 = \mathbf{u}_{n^1}^1, \qquad \mathbf{u}_{n^1}^1 = \mathbf{u}_0^2, \qquad \mathbf{u}_{n^1+1}^1 = \mathbf{u}_1^2, \tag{2.25}$$

and

$$\mathcal{G}_1^{\downarrow,L}(\mathbf{u}_0^L, \mathbf{u}_1^L) + \mathcal{N}_{n^L}^L \mathbf{f}^L = \mathbf{u}_1^L, \qquad \mathbf{u}_{n^{L-1}}^{L-1} = \mathbf{u}_0^L, \qquad \mathbf{u}_{n^{L-1}+1}^{L-1} = \mathbf{u}_1^L. \tag{2.26}$$

This was we referred to as $\underline{\mathbf{M}}\mathbf{u} = \underline{\mathbf{f}}$ in (2.11), and whose structure is depicted in Fig. 3.1 *(left)*.

Finally, the online stage of the algorithm is summarized in Alg. 1.

ALGORITHM 1. *Online computation using the SIE reduction*

11

```
 1: function  u = HELMHOLTZ SOLVER( f )
 2:     for  ℓ = 1 : L  do
 3:         fℓ = fχΩℓ                                    ▷ partition the source
 4:     end for
 5:     for  ℓ = 1 : L  do
 6:         𝒩ℓfℓ = (Hℓ)⁻¹fℓ                            ▷ solve local problems (parallel)
 7:     end for
 8:     f = (𝒩¹ₙ₁f¹, 𝒩²₁f², 𝒩²ₙ₂f², ..., 𝒩ᴸ₁fᴸ)ᵗ     ▷ form r.h.s. for the integral system
 9:     u = (M)⁻¹ f                                     ▷ solve (2.11) for the traces
10:     for  ℓ = 1 : L  do
11:         uℓⱼ = 𝒢↑,ℓⱼ(uℓₙℓ, uℓₙℓ₊₁) + 𝒢↓,ℓⱼ(uℓ₀, uℓ₁) + 𝒩ℓⱼfℓ   ▷ reconstruct local solutions
12:     end for
13:     u = (u¹, u², ..., uᴸ⁻¹, uᴸ)ᵗ                   ▷ concatenate the local solutions
14: end function
```

3. The method of polarized traces. In this section, we review succinctly the formulation of the method of polarized traces at an algebraic level; for further details see [101]. From Alg. 1 we can observe that the local solves (*lines* 5-7 in Alg. 1) and the reconstruction (*lines* 10-12 in Alg. 1) can be performed concurrently; the only sequential bottleneck is the solution of (2.11) (*line* 9 in Alg. 1). For the sake of clarity we use matrices to explain the preconditioner; however, all the operations can be performed in a matrix-free fashion, as it will be explained in Section 4.1.

The method of polarized traces was developed to solve (2.11) efficiently. The method utilizes an extended equivalent SIE formulation which relies on :

- a decomposition of the wave-field at the interfaces in two components, up-going and down-going;
- integral relations to close the new extended system;
- a permutation of the unknowns to obtain an easily preconditionable system via classical matrix splitting (see Section 4.2.2 of [84]).

Following the notation introduced in Section 2.2, the resulting extended system (see Section 3.5 in [103]) can be written as

$$\mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow},\mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_1^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow},\mathbf{u}_1^{\ell,\downarrow}) + \mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\downarrow},\mathbf{u}_{n^\ell+1}^{\ell,\downarrow}) + \mathcal{N}_1^\ell \mathbf{f}^\ell = \mathbf{u}_1^{\ell,\uparrow} + \mathbf{u}_1^{\ell,\downarrow}, \quad (3.1)$$

$$\mathcal{G}_{n^\ell}^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\uparrow},\mathbf{u}_1^{\ell,\uparrow}) + \mathcal{G}_{n^\ell}^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow},\mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_{n^\ell}^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow},\mathbf{u}_1^{\ell,\downarrow}) + \mathcal{N}_{n^\ell}^\ell \mathbf{f}^\ell = \mathbf{u}_{n^\ell}^{\ell,\uparrow} + \mathbf{u}_{n^\ell}^{\ell,\downarrow}, \quad (3.2)$$

$$\mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow},\mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_0^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow},\mathbf{u}_1^{\ell,\downarrow}) + \mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\downarrow},\mathbf{u}_{n^\ell+1}^{\ell,\downarrow}) + \mathcal{N}_0^\ell \mathbf{f}^\ell = \mathbf{u}_0^{\ell,\uparrow}, \quad (3.3)$$

$$\mathcal{G}_{n^\ell+1}^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\uparrow},\mathbf{u}_1^{\ell,\uparrow}) + \mathcal{G}_{n^\ell+1}^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow},\mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_{n^\ell+1}^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow},\mathbf{u}_1^{\ell,\downarrow}) + \mathcal{N}_{n^\ell+1}^\ell \mathbf{f}^\ell = \mathbf{u}_{n^\ell}^{\ell,\downarrow}, \quad (3.4)$$

for $\ell = 1, ..., L$; or equivalently

$$\underline{\underline{\mathbf{M}}}\,\underline{\mathbf{u}} = \underline{\mathbf{f}}, \qquad \underline{\mathbf{u}} = \begin{pmatrix} \underline{\mathbf{u}}^\downarrow \\ \underline{\mathbf{u}}^\uparrow \end{pmatrix}; \qquad\qquad (3.5)$$

where we write

$$\underline{\mathbf{u}}^\downarrow = \left(\mathbf{u}_{n^1}^{\downarrow,1}, \mathbf{u}_{n^1+1}^{\downarrow,1}, \mathbf{u}_{n^2}^{\downarrow,2}, ..., \mathbf{u}_{n^{L-1}}^{\downarrow,L-1}, \mathbf{u}_{n^{L-1}+1}^{\downarrow,L-1}\right)^t, \qquad\qquad (3.6)$$

$$\underline{\mathbf{u}}^\uparrow = \left(\mathbf{u}_0^{\uparrow,2}, \mathbf{u}_1^{\uparrow,2}, \mathbf{u}_0^{\uparrow,3}, ..., \mathbf{u}_0^{\uparrow,L}, \mathbf{u}_1^{\uparrow,L}\right)^t, \qquad\qquad (3.7)$$

to define the components of the polarized wavefields, and $\underline{\mathbf{u}}^\downarrow + \underline{\mathbf{u}}^\uparrow = \underline{\mathbf{u}}$. The indices and the arrows are chosen such that they reflect the propagation direction. For

example, $\mathbf{u}_{n^1}^{\downarrow,\ell}$ represents the wavefield leaving the layer ℓ at its bottom, i.e propagating downwards and sampled at the bottom of the layer.
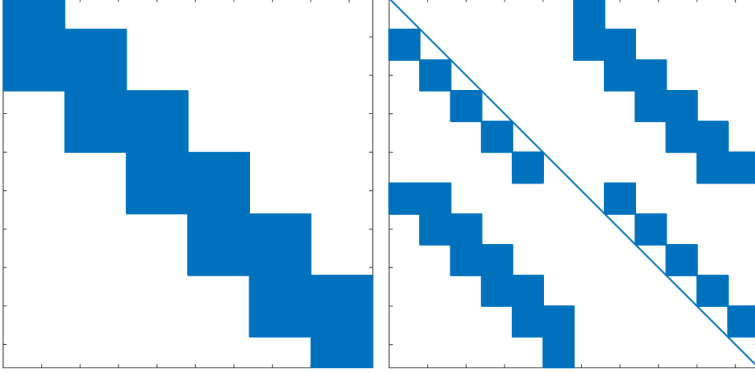

FIG. 3.1. *Sparsity pattern of the SIE matrix in* (2.11) *(left), and the polarized SIE matrix in* (3.8) *(right)* .

After a permutation of the entries (see Section 3.5 in [103], in particular Fig. 5), and some basic algebraic operations, the matrix in (3.5) takes the form

$$\underline{\underline{\mathbf{M}}} = \left[\begin{array}{cc} \mathbf{D}^\downarrow & \mathbf{U} \\ \mathbf{L} & \mathbf{D}^\uparrow \end{array} \right],$$ (3.8)

where $\underline{\mathbf{D}}^\downarrow$ and $\underline{\mathbf{D}}^\uparrow$ are, respectively, block-lower triangular and block-upper triangular matrices with identity diagonal blocks, thus easily invertible using a block back-substitution (see Fig. 3.1 (right)).

Finally, the method of polarized traces seeks to solve the system in (3.5) using an iterative method, such as GMRES, coupled with an efficient preconditioner issued from a matrix splitting, which relies on the application of $(\underline{\mathbf{D}}^\downarrow)^{-1}$ and $(\underline{\mathbf{D}}^\uparrow)^{-1}$.

We point out that the blocks of $\underline{\underline{\mathbf{M}}}$ have a physical meaning. $\underline{\mathbf{D}}^\downarrow$ takes in account the waves propagating downwards, $\underline{\mathbf{D}}^\uparrow$ considers the waves propagating upwards, $\underline{\mathbf{U}}$ takes in account the reflections of waves propagating upwards and being reflected downwards, and $\underline{\mathbf{U}}$ takes in account the down-going waves reflected upwards.

3.1. Gauss-Seidel preconditioner. In this paper, we use a block Gauss-Seidel iteration as a preconditioner to solve the polarized system in (3.5) instead of the block Jacobi iteration used in [103]. The Gauss-Seidel preconditioner is given by

$$P^{\mathrm{GS}} \left(\begin{array}{c} \mathbf{v}^\downarrow \\ \mathbf{v}^\uparrow \end{array} \right) = \left(\begin{array}{c} (\underline{\mathbf{D}}^\downarrow)^{-1} \mathbf{v}^\downarrow \\ (\underline{\mathbf{D}}^\uparrow)^{-1} \left(\mathbf{v}^\uparrow - \underline{\mathbf{L}} (\underline{\mathbf{D}}^\downarrow)^{-1} \mathbf{v}^\downarrow \right) \end{array} \right),$$ (3.9)

and the Jacobi preconditioner is given by

$$P^{\mathrm{Jac}} \left(\begin{array}{c} \mathbf{v}^\downarrow \\ \mathbf{v}^\uparrow \end{array} \right) = \left(\begin{array}{c} (\underline{\mathbf{D}}^\downarrow)^{-1} \mathbf{v}^\downarrow \\ (\underline{\mathbf{D}}^\uparrow)^{-1} \mathbf{v}^\uparrow \end{array} \right).$$ (3.10)

In our experiments, solving (3.8) using GMRES, or alternatively Bi-CGstab, preconditioned with P^{GS} converges twice as fast as using P^{Jac} as a preconditioner, and the former exhibits a weaker dependence of the number of iterations for convergence with

respect to the frequency. We considered other standard preconditioners, such as symmetric successive over-relaxation (SSOR) (see section 10.2 in [84]), but they failed to yield faster convergence, while being more computationally expensive to apply.

We point out that $\underline{\mathbf{M}}$ can be partitioned in smaller blocks. In that case, we can recover the X and NX sweeps, see Table 3 of [89], when we precondition the system with a Jacobi or Gauss-Seidel iteration respectively.
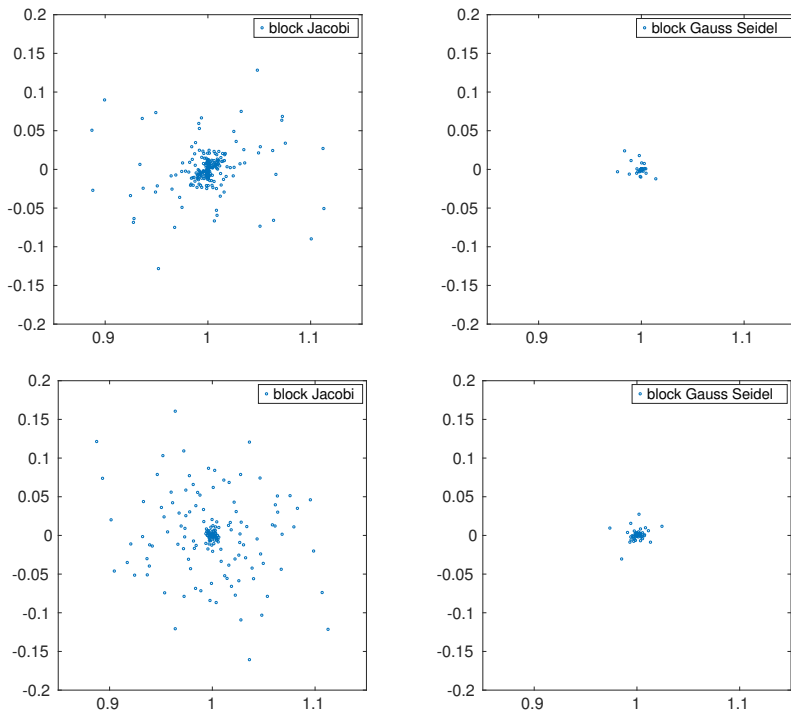

FIG. 3.2. *Eigenvalues for the preconditioned polarized systems using the block Jacobi (left) and the block Gauss-Seidel (right) preconditioner, using the BP2004 model [11] with $L = 5$, $npml = 10$, and $\omega = 34\pi$ (top row) and $\omega = 70\pi$ (bottom row).*

Fig. 3.2 depicts the eigenvalues for $\underline{\mathbf{M}}$ preconditioned with P^{GS} and P^{Jac}. We can observe that for P^{GS} the eigenvalues are more clustered and there exist fewer outliers. There exists extensive numerical evidence that indicates that a more tight clustering of the spectrum away from zero can be related to a fewer number of iteration needed to convergence[12] thus explaining the faster convergence of the GMRES iterations preconditioned with P^{Jac}.

The system in (3.5) is solved using GMRES preconditioned with P^{GS}. Moreover, as in [103] one can use an adaptive $\mathcal{H}$-matrix fast algorithms for the application of integral kernels, which in this case, are used for the solution of the local problems defined in each layer.

4. Nested solver. The main drawback of the method of polarized traces is its offline precomputation that involves computing, storing, and compressing the interface-

---

[12]If the preconditioned systems were represented by normal matrices, then from Theorem 35.2 of [91] the clustering of the eigenvalues would explain the fewer number of iteration needed to convergence. For a more extensive treatment of non-normal matrices, see [54] and references therein.

to-interface Green's functions needed to assemble $\underline{\underline{\mathbf{M}}}$. In 3D this approach would become impractical given the sheer size of the resulting matrices. To alleviate this issue, we present an equivalent matrix-free approach that relies on local solves with sources at the interfaces between layers.
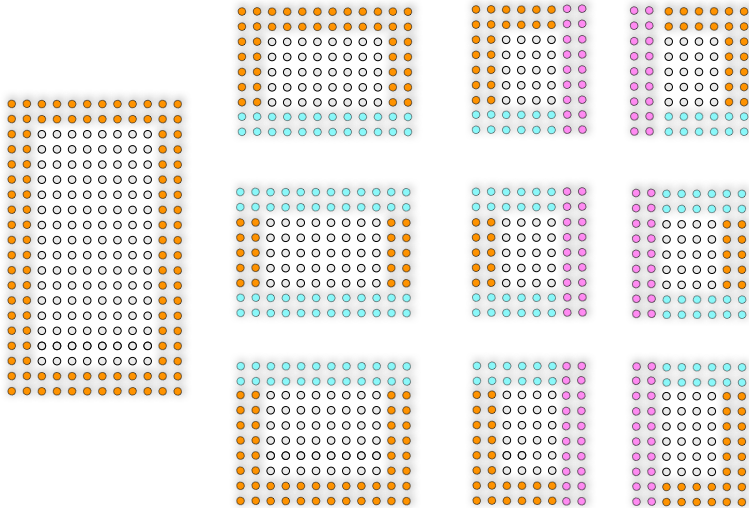

FIG. 4.1. *Nested Decomposition in cells. The orange grid-points represent the PML for the original problem, the light-blue represent the artificial PML between layers, and the pink grid-points represent the artificial PML between cells in the same layer. Compare to Fig. 2.1.*

As it will be explained in the sequel, the matrix-free approach relies on the fact that the blocks of $\underline{\underline{\mathbf{M}}}$ (as well as the blocks of $\underline{\mathbf{D}}^{\downarrow}$ and $\underline{\mathbf{D}}^{\uparrow}$) are the restrictions of local Green's functions. Thus they can be applied via a local solve (using, for example, a multifrontal sparse direct solver such as [2, 60, 27] among many others) with sources at the interfaces. This same observation was used in [103] to reconstruct the solution in the volume (see Section 2.3, in particular (28), in [103]). However, given the iterative nature of P^{GS} (that relies on inverting $\underline{\mathbf{D}}^{\downarrow}$ and $\underline{\mathbf{D}}^{\uparrow}$ by block-backsubstitution) solving the local problems naively would incur a deterioration of the online complexity, in particular, the parallelization. This deterioration can be circumvented if we solve the local problems inside the layer via the same boundary integral strategy as in the method of polarized traces, in a nested fashion. This procedure can be written as a factorization of the Green's integral in block-sparse factors, as will be explained in Section 4.2.

The nested domain decomposition approach involves a layered decomposition in $L \sim \sqrt{P}$ layers, such that each layer is further decomposed in $L_c \sim \sqrt{P}$ cells, as shown in Fig. 4.1.

In addition to the lower online complexity achieved by the nested approach, the offline complexity is much reduced; instead of computing large Green's functions for each layer, we compute much smaller interface-to-interface operators between the interfaces of adjacent cells within each layer, resulting in a lower memory footprint.

The nested approach consists of two levels:
- the *outer solver*, which solves the global Helmholtz problem, (2.10), using the matrix-free version of the method of polarized traces to solve (2.11) at the interfaces between layers;

- and the *inner solver*, which solves the local Helmholtz problems at each layer, using an integral boundary equation to solve for the degrees of freedom a the interfaces between cells within a layer.

4.1. Matrix-free approach. We proceed to explain how to implement the method of polarized traces using the matrix-free approach. We point out that the matrix-free approach is only used in the outer SIE; we still need to assemble and factorize the local systems (or local SIE's); however, the outer SIE is never assembled.

As stated before, the backbone of the method of polarized traces is to solve (3.8) iteratively with GMRES using (3.9) as a preconditioner. In this section we explain how to apply $\underline{\underline{\mathbf{M}}}$ and P^{GS} in a matrix-free fashion; and we provide the pseudo-code for the method of polarized traces using the matrix-free approach with the local solves explicitly identified.

From (3.1), (3.2), (3.3) and (3.4), each block of $\underline{\underline{\mathbf{M}}}$ is a Green's integral, and its application to a vector is equivalent to sampling a wavefield generated by suitable sources at the boundaries. The application of the Green's integral to a vector $\underline{\mathbf{v}}$, in matrix-free approach, consists of three steps:

- from $\underline{\mathbf{v}}$ we form the sources at the interfaces,
- we perform a local direct solve inside the layer,
- and we sample the solution at the interfaces.

The precise algorithm to apply $\underline{\underline{\mathbf{M}}}$ in a matrix-free fashion is provided in Alg. 2.

ALGORITHM 2. *Application of the boundary integral matrix* $\underline{\underline{\mathbf{M}}}$

1: **function** $\underline{\mathbf{u}} = $ BOUNDARY INTEGRAL$($ $\underline{\mathbf{v}}$ $)$
2: $\qquad \widetilde{\mathbf{f}}^1 = -\delta(z_{n^1+1} - z)\mathbf{v}^1_{n^\ell} + \delta(z_{n^1} - z)\mathbf{v}^2_{n^1}$ $\qquad\qquad$ ▷ *forming equivalent sources*
3: $\qquad \mathbf{w}^1 = (\mathbf{H}^1)^{-1}\widetilde{\mathbf{f}}^1$
4: $\qquad \mathbf{u}^\ell_{n^\ell} = \mathbf{w}^\ell_{n^\ell} - \mathbf{v}^\ell_{n^\ell}$ $\qquad\qquad\qquad\qquad\qquad\qquad$ ▷ *sampling*
5: $\qquad$ **for** $\ell = 2 : L - 1$ **do**
6: $\qquad\qquad \begin{aligned} \widetilde{\mathbf{f}}^\ell = &\ \delta(z_1 - z)\mathbf{v}^{\ell-1}_{n^{\ell-1}} - \delta(z_0 - z)\mathbf{v}^\ell_1 \\ &- \delta(z_{n^\ell+1} - z)\mathbf{v}^\ell_{n^\ell} + \delta(z_{n^\ell} - z)\mathbf{v}^{\ell+1}_1 \end{aligned}$ $\quad$ ▷ *forming equivalent sources*
7: $\qquad\qquad \mathbf{w}^\ell = (\mathbf{H}^\ell)^{-1}\widetilde{\mathbf{f}}^\ell$ $\qquad\qquad\qquad\qquad\qquad$ ▷ *inner solve*
8: $\qquad\qquad \mathbf{u}^\ell_1 = \mathbf{w}^\ell_1 - \mathbf{v}^\ell_1; \qquad \mathbf{u}^\ell_{n^\ell} = \mathbf{w}^\ell_{n^\ell} - \mathbf{v}^\ell_{n^\ell}$ $\qquad$ ▷ *sampling*
9: $\qquad$ **end for**
10: $\qquad \widetilde{\mathbf{f}}^L = \delta(z_1 - z)\mathbf{v}^{L-1}_{n^{L-1}} - \delta(z_0 - z)\mathbf{v}^L_1$ $\qquad\qquad$ ▷ *forming equivalent sources*
11: $\qquad \mathbf{w}^L = (\mathbf{H}^L)^{-1}\widetilde{\mathbf{f}}^L$
12: $\qquad \mathbf{u}^L_1 = \mathbf{w}^L_1 - \mathbf{v}^L_1$ $\qquad\qquad\qquad\qquad\qquad\qquad$ ▷ *sampling*
13: **end function**

Alg. 2 can be easily generalized for $\underline{\underline{\mathbf{M}}}$. We observe that there is no data dependency within the for loop, which yields an embarrassingly parallel algorithm.

Matrix-free preconditioner. For the sake of clarity we present a high level description of the implementation of (3.9) using the matrix-free version.

We use the notation introduced in Section 3 (in particular, (3.6) and (3.7)) to write explicitly the matrix-free operations for the block Gauss-Seidel preconditioner in (3.9). Alg. 3 and 4 have the physical interpretation of propagating the waves across the domains, and Alg. 5 can be seen as the up-going reflections generated by a down-going wave field. The following algorithms can be easily derived from Section 3.5 in [103].

ALGORITHM 3. *Downward sweep, application of* $(\underline{\mathbf{D}}^{\downarrow})^{-1}$

1: **function** $\underline{\mathbf{u}}^{\downarrow} = $ DOWNWARD SWEEP$($ $\underline{\mathbf{v}}^{\downarrow}$ $)$

2:    $\mathbf{u}_{n^1}^{\downarrow,1} = -\mathbf{v}_{n^1}^{\downarrow,1}$ ▷ *invert the diagonal block*

3:    $\mathbf{u}_{n^1+1}^{\downarrow,1} = -\mathbf{v}_{n^1+1}^{\downarrow,1}$

4:    **for** $\ell = 2 : L - 1$ **do**

5:        $\mathbf{w}^\ell = (\mathbf{H}^\ell)^{-1}\left[\delta(z_0 - z)\mathbf{u}_{n^{\ell-1}+1}^{\downarrow,\ell-1} - \delta(z_1 - z)\mathbf{u}_{n^{\ell-1}}^{\downarrow,\ell-1}\right]$ ▷ *inner solve*

6:        $\mathbf{u}_{n^\ell}^{\downarrow,\ell} = \mathbf{w}_{n^\ell} - \mathbf{v}_{n^\ell}^{\downarrow,\ell}$ ▷ *sample the wavefield and subtract the r.h.s.*

7:        $\mathbf{u}_{n^\ell+1}^{\downarrow,\ell} = \mathbf{w}_{n^\ell+1} - \mathbf{v}_{n^\ell+1}^{\downarrow,\ell}$ ▷ *sample the wavefield and subtract the r.h.s.*

8:    **end for**

9:    $\underline{\mathbf{u}}^\downarrow = \left(\mathbf{u}_{n^1}^{\downarrow,1}, \mathbf{u}_{n^1+1}^{\downarrow,1}, \mathbf{u}_0^{\downarrow,2}, ..., \mathbf{u}_0^{\downarrow,L-1}, \mathbf{u}_1^{\downarrow,L-1}\right)^t$

10: **end function**

ALGORITHM 4. *Upward sweep, application of* $(\underline{\mathbf{D}}^\uparrow)^{-1}$

1: **function** $\underline{\mathbf{u}}^\uparrow = $ UPWARD SWEEP($\underline{\mathbf{v}}^\uparrow$)

2:    $\mathbf{u}_0^{\uparrow,L} = -\mathbf{v}_0^{\uparrow,L}$ ▷ *invert the diagonal block*

3:    $\mathbf{u}_1^{\uparrow,L} = -\mathbf{v}_1^{\uparrow,L}$

4:    **for** $\ell = L - 1 : 2$ **do**

5:        $\mathbf{w}^\ell = (\mathbf{H}^\ell)^{-1}\left[-\delta(z_{n^\ell+1} - z)\mathbf{u}_1^{\uparrow,\ell-1} + \delta(z_{n^\ell} - z)\mathbf{u}_0^{\uparrow,\ell-1}\right]$ ▷ *inner solve*

6:        $\mathbf{u}_1^{\uparrow,\ell} = \mathbf{w}_1^\ell - \mathbf{v}_1^{\uparrow,\ell}$ ▷ *sample the wavefield and subtract the r.h.s.*

7:        $\mathbf{u}_0^{\uparrow,\ell} = \mathbf{w}_0^\ell - \mathbf{v}_0^{\uparrow,\ell}$ ▷ *sample the wavefield and subtract the r.h.s.*

8:    **end for**

9:    $\underline{\mathbf{u}}^\uparrow = \left(\mathbf{u}_0^{\uparrow,1}, \mathbf{u}_1^{\uparrow,1}, \mathbf{u}_{n^2}^{\uparrow,2}, ..., \mathbf{u}_{n^{L-1}}^{\uparrow,L-1}, \mathbf{u}_{n^{L-1}+1}^{\uparrow,L-1}\right)^t$

10: **end function**

ALGORITHM 5. *Upward Reflections, application of* $\underline{\mathbf{L}}$

1: **function** $\underline{\mathbf{u}}^\uparrow = $ UPWARD REFLECTIONS($\underline{\mathbf{v}}^\downarrow$)

2:    **for** $\ell = 2 : L - 1$ **do**

3:        $\mathbf{f}^\ell = \begin{aligned}&\delta(z_1 - z)\mathbf{v}_0^{\downarrow,\ell} - \delta(z_0 - z)\mathbf{v}_1^{\downarrow,\ell}\\&-\delta(z_{n^\ell+1} - z)\mathbf{v}_1^{\downarrow,\ell+1} + \delta(z_{n^\ell} - z)\mathbf{v}_0^{\downarrow,\ell+1}\end{aligned}$

4:        $\mathbf{w}^\ell = (\mathbf{H}^\ell)^{-1}\mathbf{f}^\ell$ ▷ *inner solve*

5:        $\mathbf{u}_1^{\uparrow,\ell} = \mathbf{w}_1^\ell - \mathbf{v}_1^{\downarrow,\ell}$ ▷ *sample the wavefield and subtract the identity*

6:        $\mathbf{u}_0^{\uparrow,\ell} = \mathbf{w}_0^\ell$ ▷ *sample the wavefield*

7:    **end for**

8:    $\mathbf{f}^L = \delta(z_1 - z)\mathbf{v}_0^{\uparrow,L} - \delta(z_0 - z)\mathbf{v}_1^{\uparrow,L}$

9:    $\mathbf{w}^L = (\mathbf{H}^L)^{-1}\mathbf{f}^L$ ▷ *local solve*

10:   $\mathbf{u}_1^{\uparrow,L} = \mathbf{w}_1^L - \mathbf{v}_1^{\downarrow,L}$ ▷ *sample the wavefield and subtract the identity*

11:   $\mathbf{u}_0^{\uparrow,L} = \mathbf{w}_0^L$ ▷ *sample the wavefield*

12:   $\underline{\mathbf{u}}^\uparrow = \left(\mathbf{u}_0^{\uparrow,2}, \mathbf{u}_1^{\uparrow,2}, \mathbf{u}_{n^2}^{\uparrow,3}, ..., \mathbf{u}_{n^{L-1}}^{\uparrow,L-1}, \mathbf{u}_{n^L+1}^{\uparrow,L}\right)^t$

13: **end function**

We observe that the for loop in line 2-7 in Alg. 5 is completely parallel. On the other hand, in Alg. 3 and 4, the data dependency within the for loop forces the algorithm to run sequentially. The most expensive operation is the inner solve performed locally at each layer. We will argue in Section 4.2 that using a nested approach, with an appropriate reduction of the degrees of freedom, we can obtain a highly efficient inner solve, which yields a fast application of the preconditioner.

Matrix-free solver. We provide the full algorithm of the matrix-free solver using the method of polarized traces coupled with the Gauss-Seidel preconditioner. The main difference with the original method of polarized traces in [103] is that we

use Algs. 2, 3, 4, and 5 within the GMRES iteration (line 8 of Alg. 6) instead of compressed matrix-vector multiplications.

ALGORITHM 6. *Matrix-free polarized traces solver*

1: **function** $\mathbf{u} = $ MATRIX-FREE SOLVER$(\ \mathbf{f}\ )$
2:      **for** $\ell = 1 : L$ **do**
3:          $\mathbf{f}^\ell = \mathbf{f}\chi_{\Omega^\ell}$ $\triangleright$ *partition the source*
4:          $\mathbf{w}^\ell = \left(\mathbf{H}^\ell\right)^{-1}\left(\mathbf{f}^\ell\right)$ $\triangleright$ *solve local problems*
5:      **end for**
6:      $\underline{\mathbf{f}} = \left(\mathbf{w}^1_{n^1}, \mathbf{w}^2_1, ..., \mathbf{w}^L_1\right)^t; \qquad \underline{\mathbf{f}}_0 = \left(\mathbf{w}^1_{n^1+1}, \mathbf{w}^2_0, ..., \mathbf{w}^L_0\right)^t$
7:      $\underline{\underline{\mathbf{f}}} = \begin{pmatrix} \underline{\mathbf{f}} \\ \underline{\mathbf{f}}_0 \end{pmatrix}$ $\triangleright$ *form the r.h.s. for the polarized integral system*
8:      $\begin{pmatrix} \underline{\mathbf{u}}^\downarrow \\ \underline{\mathbf{u}}^\uparrow \end{pmatrix} = \underline{\underline{\mathbf{u}}} = \left(P^{GS}\underline{\underline{\mathbf{M}}}\right)^{-1} P^{GS}\underline{\underline{\mathbf{f}}}$ $\triangleright$ *solve using GMRES*
9:      $\underline{\mathbf{u}} = \underline{\mathbf{u}}^\uparrow + \underline{\mathbf{u}}^\downarrow$ $\triangleright$ *add the polarized components*
10:      $\widetilde{\mathbf{f}}^1 = \mathbf{f}^1 - \delta(z_{n^1+1} - z)\mathbf{u}^1_{n^1} + \delta(z_{n^1} - z)\mathbf{u}^2_1$ $\triangleright$ *reconstruct local solutions*
11:      $\mathbf{u}^1 = \left(\mathbf{H}^1\right)^{-1}\left(\widetilde{\mathbf{f}}^1\right)$
12:      **for** $\ell = 2 : L - 1$ **do**
13:          $\begin{aligned} \widetilde{\mathbf{f}}^\ell = \quad & \mathbf{f}^\ell + \delta(z_1 - z)\mathbf{u}^{\ell-1}_{n^{\ell-1}} - \delta(z_0 - z)\mathbf{u}^\ell_1 \\ & -\delta(z_{n^\ell+1} - z)\mathbf{u}^\ell_{n^\ell} + \delta(z_{n^\ell} - z)\mathbf{u}^{\ell+1}_1 \end{aligned}$
14:          $\mathbf{u}^\ell = \left(\mathbf{H}^\ell\right)^{-1}\left(\widetilde{\mathbf{f}}^\ell\right)$
15:      **end for**
16:      $\widetilde{\mathbf{f}}^L = \mathbf{f}^L + \delta(z_1 - z)\mathbf{u}^{L-1}_{n^{L-1}} - \delta(z_0 - z)\mathbf{u}^L_1$
17:      $\mathbf{u}^L = \left(\mathbf{H}^L\right)^{-1}\left(\widetilde{\mathbf{f}}^L\right)$
18:      $\mathbf{u} = (\mathbf{u}^1, \mathbf{u}^2, \dots, \mathbf{u}^{L-1}, \mathbf{u}^L)^t$ $\triangleright$ *concatenate the local solutions*
19: **end function**

The matrix-free solver in Alg. 6 has three stages :

- *lines 2-7:* preparation of the r.h.s. for the outer polarized integral system, this can be done concurrently within each layer;
- *lines 8-9:* solve for the traces at the interfaces between layers, using preconditioned GMRES, and applying $\underline{\underline{\mathbf{M}}}$ and the preconditioner via the matrix-free approach with Algs. 2, 3, 4 and 5;
- *lines 10-18:* reconstruction of the solution inside the volume at each layer, which can be performed independently of the other layers.

4.2. Nested inner and outer solver. In the presentation of the matrix-free solver (Alg. 6), we have extensively relied on the assumption that the inner systems $\mathbf{H}^\ell$ can be solved efficiently in order to apply the Green's integrals fast. In this section we describe the algorithms to compute the solutions to the inner, or local, systems efficiently, and then we describe how the outer solver calls the inner solver.

From the analysis of the rank of the off-diagonal blocks of the Green's functions we know that the Green's integrals can be compressed in a way that results in a fast application in $\mathcal{O}(n^{3/2})$ time (see Section 5 in [103]), but this approach requires precomputation and storage of the Green's functions. The matrix-free approach in Alg. 6 does not need expensive precomputations, but it would naively perform a direct solve in the volume (inverting $\mathbf{H}^\ell$), resulting in an application of the Green's integral in $\mathcal{O}(N/L)$ complexity up to logarithmic factors (assuming that a good direct method is used at each layer). This becomes problematic when applying the preconditioner, which involves $\mathcal{O}(L)$ sequential applications of the Green's integrals

as Algs. 3 and 4 show. This means that the unaided application of the preconditioner using the matrix-free approach would result in an algorithm with linear online complexity, in particular, the method would behave similarly to a sweeping-like preconditioner. The nested strategy (that we present below) mitigates this issue resulting in a lower $\mathcal{O}(L_c(n/L)^{3/2})$ (up to logarithmic factors) complexity for the application of each Green's integral, where L_c is the number of cells per layer.
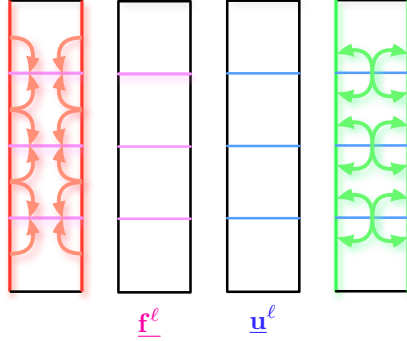

FIG. 4.2.   *Sketch of the application of the Green's functions using a nested approach. The sources are in red (left) and the sampled field in green (right). The application uses the inner boundaries as proxies to perform the solve.*

We follow the matrix-free approach of Alg. 6, but instead of a direct solver to invert $\mathbf{H}^\ell$, we use a nested solver, i.e., we use the same reduction used in Ω to each layer Ω^ℓ. We reduce the local problem at each layer to solving a discrete integral system analog to (2.11) with a layered decomposition in the transverse direction given by

$$\underline{\mathbf{M}}^\ell \underline{\mathbf{u}}^\ell = \underline{\mathbf{f}}^\ell, \qquad \text{for } \ell = 1, .., L_c. \tag{4.1}$$

We suppose that we have $L_c \sim L \sim \sqrt{P}$ cells in each layer.

The nested solver uses the inner boundaries, or interfaces between cells, as proxies to perform the local solve inside the layer efficiently. The efficiency can be improved when the inner solver is used in the applications of the Green's integral within the preconditioner. The improved efficiency stems from the localization of the sources, and the sampling of the solution on the interfaces, which allows us to precompute and compress some for the operations. In that case, the application of the Green's integral can be decomposed into three steps:

- using precomputed Green's functions at each cell we evaluate the wavefield generated from the sources to form $\underline{\mathbf{f}}^\ell$ (from red to pink in Fig. 4.2 (left)); this operation can be represented by a sparse block matrix $\underline{\mathbf{M}}_f^\ell$;
- we solve (4.1) to obtain $\underline{\mathbf{u}}^\ell$ (from pink to blue in Fig. 4.2 (right));
- finally, we use the Green's representation formula to sample the wavefield at the interfaces (from blue to green in Fig. 4.2), this operation is represented by another sparse-block matrix $\underline{\mathbf{M}}_u^\ell$.

Using the definition of the incomplete integrals in Section 2.2 the algorithm de-

scribed above leads to the factorization

$$
\begin{bmatrix}
\mathcal{G}_0^{\downarrow,\ell}(\mathbf{v}_0,\mathbf{v}_1) + \mathcal{G}_0^{\uparrow,\ell}(\mathbf{v}_{n^\ell},\mathbf{v}_{n^\ell+1}) \\
\mathcal{G}_1^{\downarrow,\ell}(\mathbf{v}_0,\mathbf{v}_1) + \mathcal{G}_0^{\uparrow,\ell}(\mathbf{v}_{n^\ell},\mathbf{v}_{n^\ell+1}) \\
\mathcal{G}_{n^\ell}^{\downarrow,\ell}(\mathbf{v}_0,\mathbf{v}_1) + \mathcal{G}_{n^\ell}^{\uparrow,\ell}(\mathbf{v}_{n^\ell},\mathbf{v}_{n^\ell+1}) \\
\mathcal{G}_{n^\ell+1}^{\downarrow,\ell}(\mathbf{v}_0,\mathbf{v}_1) + \mathcal{G}_{n^\ell+1}^{\uparrow,\ell}(\mathbf{v}_{n^\ell},\mathbf{v}_{n^\ell+1})
\end{bmatrix}
= \underline{\mathbf{M}}_u^\ell \left(\underline{\mathbf{M}}^\ell\right)^{-1} \underline{\mathbf{M}}_f^\ell \cdot
\begin{bmatrix}
\mathbf{v}_0 \\
\mathbf{v}_1 \\
\mathbf{v}_{n^\ell} \\
\mathbf{v}_{n^\ell+1}
\end{bmatrix}, \quad (4.2)
$$

in which the blocks of $\underline{\mathbf{M}}_f^\ell$ and $\underline{\mathbf{M}}_u^\ell$ are dense, but compressible in partitioned low rank (PLR)[13] form.

Algorithms. We now provide the algorithms in pseudo-code for the the inner solver, Alg. 7, and we provide the necessary modifications to Alg. 6 for the outer solve. In addition, we provide a variant of the inner solver that is crucial to obtain the online complexity mentioned at the beginning of the paper (i.e. $\mathcal{O}(N/P)$ provided that $P = \mathcal{O}(N^{1/5})$).

In order to reduce the notational burden, we define the inner solve using the same notation as before. We suppose that each layer Ω^ℓ is decomposed in L_c cells, noted $\{\Omega^{\ell,c}\}_{c=1}^{L_c}$. We extend all the definitions from the matrix-free solver to the inner solver, by indexing the operations by ℓ and c, in which, ℓ stands for the layer-index and c for the cell-index within the layer.

For each Ω^ℓ, we apply the variable swap $\widetilde{\mathbf{x}} = (z,x)$, which is noted by $\mathcal{R}$ such that $\mathcal{R}^2 = I$. Under the variable swap, we can decompose Ω^ℓ in L_c layers, or cells, $\{\Omega^{\ell,c}\}_{c=1}^{L_c}$ to which we can apply the machinery of the boundary integral reduction at the interfaces between cells. The resulting algorithm has the same structure as before. The variable swap is a suitable tool that allows us to reuse to great extent the notation introduced in [103]. Numerically, the variable swap just introduced is implemented by transposing the matrices that represent the different wavefields.

ALGORITHM 7. *Inner Solve, for applying* $(\mathbf{H}^\ell)^{-1}$ *in Algs. 3, 4 and 5*

1: **function** $\mathbf{w} = $ INNER SOLVER$^\ell($ $\mathbf{f}^\ell$ $)$
2:     $\mathbf{g}^\ell = \mathcal{R} \circ \mathbf{f}^\ell$ ▷ *variable swap*
3:     **for** $c = 1 : L_c$ **do**
4:         $\mathbf{g}^{\ell,c} = \mathbf{g}\chi_{\Omega^{\ell,c}}$ ▷ *partition the source*
5:     **end for**
6:     **for** $c = 1 : L_c$ **do**
7:         $\mathcal{N}^{\ell,c}\mathbf{g}^{\ell,c} = (\mathbf{H}^{\ell,c})^{-1}\mathbf{g}^{\ell,c}$ ▷ *solve local problems*
8:     **end for**
9:     $\underline{\mathbf{g}}^\ell = \left(\mathcal{N}_{n^1}^{\ell,1}\mathbf{g}^{\ell,1}, \mathcal{N}_1^{\ell,2}\mathbf{g}^{\ell,2}, \mathcal{N}_{n^2}^{\ell,2}\mathbf{g}^{\ell,2}, \ldots, \mathcal{N}_1^{\ell,L_c}\mathbf{g}^{\ell,L_c}\right)^t$ ▷ *form r.h.s.*
10:     $\underline{\mathbf{v}}^\ell = \left(\underline{\mathbf{M}}^\ell\right)^{-1}\underline{\mathbf{g}}^\ell$ ▷ *solve for the traces* (4.1)
11:     **for** $c = 1 : L_c$ **do**   ▷ *local reconstruction*
12:         $\mathbf{v}_j^{\ell,c} = \mathcal{G}_j^{\uparrow,\ell,c}(\mathbf{v}_{n^\ell}^{\ell,c}, \mathbf{v}_{n^\ell+1}^{\ell,c}) + \mathcal{G}_j^{\downarrow,\ell,c}(\mathbf{v}_0^{\ell,c}, \mathbf{v}_1^{\ell,c}) + \mathcal{N}_j^{\ell,c}\mathbf{g}^{\ell,c}$
13:     **end for**
14:     $\mathbf{v}^\ell = (\mathbf{v}^{\ell,1}, \mathbf{v}^{\ell,2}, \ldots, \mathbf{v}^{\ell,L_c-1}, \mathbf{v}^{\ell,L_c})^t$ ▷ *concatenate the local solutions*
15:     $\mathbf{w} = \mathcal{R} \circ \mathbf{v}^\ell$ ▷ *variable swap*
16: **end function**

To obtain the nested solver, we modify Alg. 6, and the algorithm it calls, by replacing $(\mathbf{H}^\ell)^{-1}$ by the inner solver.

---

[13]A PLR matrix is a $\mathcal{H}$-matrix obtained by an adaptive dyadic partitioning and multilevel compression, see Section 3 in [103]

- If the support of the source is the whole layer and the wavefield is required in the volume, we use the inner solve as prescribed in Alg. 7 without modifications. In Alg. 6 we modify lines $4, 11, 14$, and 17, in which $(\mathbf{H}^\ell)^{-1}$ is replaced by Alg. 7.
- If the source term is concentrated at the interfaces between layers, and the wavefield is needed only at the interfaces, we reduce the computational cost by using a slight modification of Alg. 7, according to (4.2). In this variant, the local solves in line 7 of Alg. 7 (which is performed via a LU back-substitution) and the reconstruction (lines 11 to 15 in Alg. 7), are replaced by precomputed operators as it was explained above. Within the GMRES loop (line 10 in Alg. 6), we replace $(\mathbf{H}^\ell)^{-1}$ with the variant of Alg. 7 following (4.2), in line 5 of Alg. 3; line 4 of Alg. 4, and lines 4 and 9 of Alg. 5.

The choice of algorithm to solve (4.1) and to apply the Green's integrals dictates the scaling of the offline complexity and the constant of the online complexity. We can either use the method of polarized traces or the compressed-block LU solver, which are explained below.

4.2.1. Nested polarized traces. To efficiently apply the Green's integrals using Alg. 7, we need to solve (4.1) efficiently. One alternative is to use the method of polarized traces in a recursive fashion to solve the system at each layer. We call this approach the method of nested polarized traces. Following [103] this approach has the same empirical scalings, at the inner level, as those found in [103] when the blocks are compressed in PLR form.

Although, as it will be explained Section 4.3, in this case the complexity is lower, we have to iterate inside each layer to solve each system, which produces large constants for the application of the Green's integrals in the online stage.

4.2.2. Inner compressed-block LU. An alternative to efficiently apply the Green's integrals via Alg. 7, is to use the compressed-block LU solver (see Chapter 3 in [101]) to solve (4.1). Given the banded structure of $\underline{\mathbf{M}}^\ell$ (see Fig. 3.1 (left)), we perform a block LU decomposition without pivoting. The resulting LU factors are block sparse and tightly banded. We have the factorization

$$\underline{\mathbf{M}}^\ell = \underline{\mathbf{L}}^\ell \, \underline{\mathbf{U}}^\ell, \tag{4.3}$$

which leads to

$$\mathcal{G}^\ell = \underline{\mathbf{M}}_u^\ell (\underline{\mathbf{U}}^\ell)^{-1} (\underline{\mathbf{L}}^\ell)^{-1} \underline{\mathbf{M}}_f^\ell, \tag{4.4}$$

in which $\mathcal{G}^\ell$ represents the linear operator at the left-hand-side of (4.2). Following Section 3 in [101], $(\underline{\mathbf{U}}^\ell)^{-1}(\underline{\mathbf{L}}^\ell)^{-1}$ can be done in the same complexity as the nested polarized traces, at the price of a more thorough precomputation. The improved complexity is achieved by inverting the diagonal blocks of the LU factors, thus reducing $(\underline{\mathbf{U}}^\ell)^{-1}$ and $(\underline{\mathbf{L}}^\ell)^{-1}$ to a sequence of matrix-vector multiplications that are further accelerated by compressing the matrices in PLR form.

The main advantage of the inner compressed solver with respect to using the method of polarized traces in the layer solve, is that we do not need to iterate and the system to solve is half the size. Therefore, the online constants are much lower than using the method of polarized traces for the inner solve.

4.3. Complexity. Table 4.1 summarizes the complexities and number of processors at each stage of the nested polarized traces method in Section 4.2.1.

Step	N_{nodes} active	Complexity per node	Communication
LU factorizations	$\mathcal{O}(P)$	$\mathcal{O}\left((N/P + \log(N))^{3/2}\right)$	$\mathcal{O}(1)$
Green's functions	$\mathcal{O}(P)$	$\mathcal{O}\left((N/P + \log(N))^{3/2}\right)$	$\mathcal{O}(1)$
Local solves	$\mathcal{O}(P)$	$\mathcal{O}\left(N/P + \log(N)^2\right)$	$\mathcal{O}(1)$
Sweeps	1	$\mathcal{O}(P(N/P + \log(N)^2)^\alpha)$	$\mathcal{O}(PN^{1/2})$
Reconstruction	$\mathcal{O}(P)$	$\mathcal{O}\left(N/P + \log(N)^2\right)$	$\mathcal{O}(1)$

TABLE 4.1

Number of nodes active, complexity, and communication cost of the different steps of the pre-conditioner, in which α depends on the compression of the local matrices, thus on the scaling of the frequency with respect to the number of unknowns. The value of α depends on the compression rate of the discrete integral operator, which depends on the scaling of ω, N, and the maximum rank of the blocks (see Section 5.3 of [103], in particular, Table 4). Typically $\alpha = 3/4$.

The runtimes and complexities presented in Table 4.1 use the following setup: We suppose that we have available $\mathcal{O}(P)$ nodes, and they are organized following the domain decomposition in Fig. 4.1. In particular, we have $L \sim \sqrt{P}$ layers, $\mathcal{O}(\sqrt{P})$ nodes per layer, and $\mathcal{O}(1)$ nodes per cell, i.e., $L_c \sim \sqrt{P}$ and $L \cdot L_c \sim P$. We suppose that the factorized matrices, and compressed Green's functions are local to the nodes associated to a cell. The cells communicate only with theirs neighbors induced by the topology of the decomposition. For simplicity we do not count the logarithmic factors from the nested dissection; however, we consider the logarithmic factors coming from the extra degrees of freedom in the PML[14].

4.3.1. Offline Complexity. The offline stage is composed of the LU factorizations at each cell containing $\mathcal{O}(N/P + \log(N))$ points, the computation of the Green's function that involves solving $\mathcal{O}(n/\sqrt{P})$ local systems in each cell, and the compression of the resulting Green's functions in PLR form. The complexities for each node are presented in From Table 4.1; furthermore, the compression of the Green's functions takes a negligible $\mathcal{O}(n/\sqrt{P} \log(n/\sqrt{P}))$ time per node (see Table 2 and Section 5.1 in [103]) using randomized methods (cf. [73]) to accelerate the compression step.

From Table 4.1, the offline stage is embarrassingly parallel at the cell level and it has an overall runtime $\mathcal{O}\left((N/P)^{3/2}\right)$, up to logarithmic factors, as stated in Table 1.1.

When using the inner compressed-block LU method, we have the same complexity but with an extra $\mathcal{O}(N^{3/2}/P)$ cost in the offline stage, making it comparable to the method of polarized traces in [103], although with a lower online complexity.

4.3.2. Online Complexity. For the online stage, the runtime of the local solves and the reconstruction in each cell is independent of the frequency and embarrassingly parallel; the runtime is then dominated by the complexity of multifrontal methods, as stated in Table 4.1.

The sweeps are however, fully sequential, as shown in Table 4.1. Moreover, given the nested nature of the preconditioner, we have two kinds of sweeps:
- the inner sweeps within a layer that are used to apply the Green's integral in the outer SIE,
- and the outer sweeps, which sweeps from layer to layer applying the Green's integrals in a matrix-free fashion; therefore, relying on the inner sweeps.

The runtime of the inner sweeps depends on the compression rate of the integral operators involved in the local SIE, which is given by α that is exponent of the

---

[14]They are more visible in the runtime scalings presented below.

empirical asymptotic complexity of the application of integral operators with respect to N. The value of α depends on the scaling between the frequency and the number of degrees of freedom per dimension. If the frequency scales as $\omega \sim \sqrt{n}$, the regime in which second order finite-differences and Q1 finite elements are expected to be quasi-optimal, we empirically obtain $\alpha = 5/8$ (see Fig. 13 in [103]); however, we assume the more conservative value $\alpha = 3/4$. The latter is in better agreement with a theoretical analysis of the ranks of the off-diagonal blocks of the Green's functions under a geometrical optics approximation (see Section 5 in [103], in particular, Table 4 and Lemma 7; for further details on the compressibility of Green's functions for smooth heterogeneous media see [35]).

In such scenario, the complexity of the application of the Green's integral at the layer level depends on the complexity of the application of each of the factors in the right-hand side of (4.2), i.e., $\underline{\mathbf{M}}_u^\ell$, $(\underline{\mathbf{M}}^\ell)^{-1}$, and $\underline{\mathbf{M}}_f^\ell$.

For the operators $\underline{\mathbf{M}}_u^\ell$ and $\underline{\mathbf{M}}_f^\ell$, we can compress theirs blocks in PLR format (see Section 5 of [103]). From (4.2) and Fig. 4.2, we can clearly see that $\underline{\mathbf{M}}_u^\ell$ and $\underline{\mathbf{M}}_f^\ell$ are integral operators with blocks involving the numerical Green's function local to each cell sampled at the interfaces. Each block can be represented by a matrix of size $(n/L + \log(n)) \times (n/L + \log(n))$[15], thus after compression it is possible to apply each block in $\mathcal{O}((n/L + \log(n))^{2\alpha})$ time. This last statement is backed from the extensive numerical experiments whose results are summarixed in Table 4 of [103], in which the complexity of the application of a compressed $n \times n$ matrix issued from the discrete Green's function, is $\mathcal{O}(n^{2\alpha})$ or, following the fact that $N = n^2$, $\mathcal{O}(N^\alpha)$. Finally, given that the operators $\underline{\mathbf{M}}_u^\ell$ and $\underline{\mathbf{M}}_f^\ell$ have L_c blocks, then they can be applied in $\mathcal{O}(L_c(n/L + \log(n))^{2\alpha})$ time.

For the application of $(\underline{\mathbf{M}}^\ell)^{-1}$, the remaining term in (4.2), we can solve (4.1) using either the compressed-block LU or the nested polarized traces in $\mathcal{O}(L_c(n/L + \log(n))^{2\alpha})$ time. We follow the same procedure as in the method of polarized traces: we build an extended local SIE, and we solve the local SIE iteratively using preconditioned GMRES. The solve is accelerated by compressing the blocks of the local SIE. Each block of the local SIE is represented by a $(n/L + \log(n)) \times (n/L + \log(n))$ matrix, thus, following Section 4 in [103], the application of each block can be performed in $\mathcal{O}((n/L + \log(n))^{2\alpha})$ time. Therefore, given that each layers has L_c cells, the application of the local SIE and the local preconditioner after compression can be performed in $\mathcal{O}(L_c(n/L + \log(n))^{2\alpha})$ time. For the case of the compressed-block LU variant a similar argument provides the same asymptotic complexity.

This yields a runtime of $\mathcal{O}(L_c(n/L + \log(n))^{2\alpha})$ for each application of the Green's integral, at the layer level, using the factorization in (4.2).

Finally, to apply the Gauss-Seidel preconditioner on the outer SIE, we need to perform outer sweeps, each requiring $\mathcal{O}(L)$ applications of the Green's integrals, resulting in a runtime of $\mathcal{O}(L \cdot L_c(n/L + \log(n))^{2\alpha})$ to solve (3.5). Using the fact that $L \sim \sqrt{P}$, $L_c \sim \sqrt{P}$, $N = n^2$ and adding the contribution of the other steps of the online stage; we have that the overall online runtime is given by $\mathcal{O}(P^{1-\alpha}N^\alpha + P\log(N)^\alpha + N/P + \log(N)^2)$. Supposing that $P = \mathcal{O}(N)$ and neglecting the logarithmic factors we have that the overall runtime is given by $\mathcal{O}(P^{1-\alpha}N^\alpha + N/P)$ as stated in Table 1.1.

Moreover, if $\alpha = 3/4$, then we have that the online complexity is $\mathcal{O}(N/P)$ (up to logarithmic factors) provided that $P = \mathcal{O}\left(N^{1/5}\right)$. The communication cost for

---

[15]The logarithmic factor comes from the fact that the traces are taken transversally to each layer, then we need to consider the extra PML nodes, which grow as $\mathcal{O}(\log n)$.
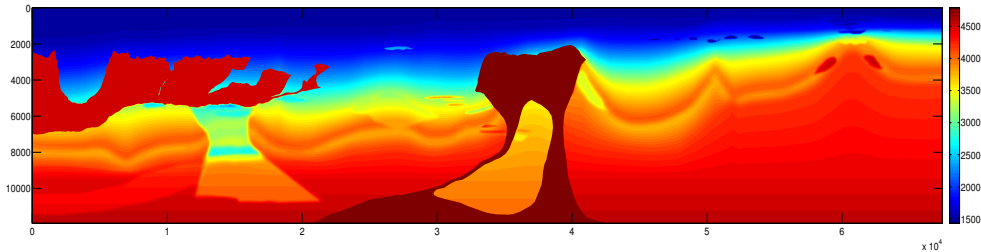
Fig. 4.3. *BP2004 geophysical benchmark model [11]*

the online part is $\mathcal{O}(nP)$, and the memory footprint is $\mathcal{O}(P^{1/4}N^{3/4} + P\log(N)^{3/4} + N/P + \log(N)^2)$, which represents an asymptotic improvement with respect to [103], in which the memory footprint is $\mathcal{O}(PN^{3/4} + N/P + \log(N)^2)$.

It was already explained in [103] why $\alpha = 3/4$ is a reasonable assumption. Empirically, α is often closer to $5/8$, but this seems to be an overly optimistic pre-asymptotic scaling. Theoretically, the case can be made for $\alpha = 3/4$ in the continous geometrical optics scenario when $G(x,y) = A_\omega(x,y)e^{i\omega\Phi(x,y)}$ for smooth $A_\omega(x,y)$ and $\Phi(x,y)$ except when $x \neq y$, and $A_\omega(x,y)$ further depending on ω in a harmless polyhomogeneous way [16]. When $\omega \sim n$, then it is easy to show (by factoring out the leading plane wave) that the largest constant-rank blocks have size $O(\sqrt{\omega}) = O(\sqrt{n})$. Further bookkeeping on the partitioned low-rank structure induced by these blocks shows that the compressed matrix-vector multiplication can be realized in $O(n^{3/2}) = O(N^{3/4})$ operations, yielding $\alpha = 3/4$. This argument is not rigorous for two reasons: (i) geometrical optics may not be a good approximation, and (ii) it does not take into account the fact that the Green's function for the discretized problem may be far from that of the PDE. If the Green's functions are not compressed, it is clear that $\alpha = 1$.

5. Numerical results. The code used for the numerical experiments was written in MATLAB, and the experiments were performed in a dual socket server with two Xeon E5-2670 and 384 GB of RAM. Given the lack of parallelism of the MATLAB implementation we only benchmark the sequential bottleneck of the online computation, which is the only non embarrassingly parallel operation. Following Table 4.1 the communication cost is asymptotically negligible, so we focus the benchmarks on the number of iterations needed for convergence, and on the compressibility of the Green's functions at the inner level.

In this section we provide numerical evidence of the following claims:
- the fast convergence of the method of polarized traces for a typical geophysical benchmark model and its agnosticity to the type of sources;
- the behavior of the preconditioner using two different matrix splittings and different Krylov iterative methods to solve the outer problem;
- the dependence of the number of iteration to convergence with respect to the frequency, the contrast of the medium, the number of subdomains and the number of layers in layered media.
- the effectiveness of the compression of the local matrices to reduce the runtime,
- the complexity reduction due to the compression of the local matrices coupled with a proper scaling of the number of subdomains.

---

[16]I.e., $A_\omega(x,y)$ has no highly oscillatory components depending on ω.

N	$\omega/2\pi$ [Hz]	$L \times L_c$	$u_0 = 0$	$u_0 = rand$
60×169	1.25	6×3	**3**	**5**
120×338	2.5	12×6	**4**	**6**
239×675	5.0	24×12	**5**	**8**
478×1349	10.0	48×24	**6**	**10**
955×2697	20.0	96×48	**6**	**12**

TABLE 5.1

Maximum number of iterations required to reduce the relative residual to 10^{-5}, using GMRES with the Gauss-Seidel preconditioner, for 50 realizations of a random source, using a zero initial guess an a randomly generated initial guess. In this case, $f = \omega/2\pi \sim n$; the number of layers and the number of cells inside each layer grows as n; the number of points in the PML scales as $\log(N)$, and the sound speed is given by the BP 2004 model (see [11]).
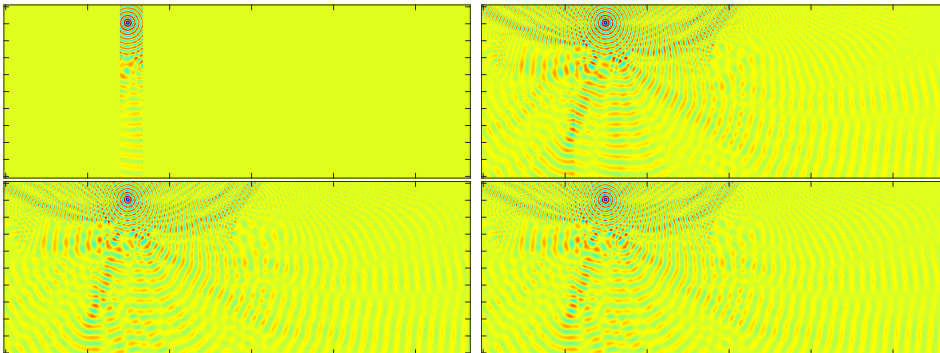

FIG. 5.1. *Two iterations of the preconditioner, from left to right and top to bottom: initial guess with only local solves; first iteration, second iteration, solution. The background model is given by the BP 2004 model [11] shown in Fig. 4.3.*

5.1. Fast convergence. Fig. 5.1 depicts the fast convergence of the method when using the BP2004 model[17] (see Fig. 4.3), point sources and a layered domain decomposition with vertical layers. After a couple of iterations the exact and approximated solutions are indistinguishable to the naked eye. Fig. 5.2 depicts the fast convergence of the method of polarized traces when using the BP2004 model as wave-speed, but with a global random source. We can observe that the method converges extremely fast, after 2 iterations the exact and approximated solutions are indistinguishable to the naked eye. Even though most of the experiments shown in this manuscript were performed using randomly located point sources, which are of paramount importance in Geophysics to approximate seismic sources, the behavior of the solver is agnostic to the form and location of the source as shown in Table 5.2, which presents the number of iterations to convergence, when using a randomly located point source, or a global source randomly generated by the `rand` function in MATLAB.

5.2. Convergence using different matrix splittings and Krylov solvers. In the prequel we have defined two different preconditioners: the block Jacobi (3.10), and block Gauss-Seidel preconditioner (3.9); which are essentially different block matrix splittings of the polarized system in (3.8). We compare the performance of the both preconditioners for two different Krylov subspace methods used to solve the po-

---

[17]The 2004 BP model is a typical benchmark model in the Geophysical community, and it includes simplified geologic features usually found in the Eastern/Central Gulf of Mexico and off-shore Angola.
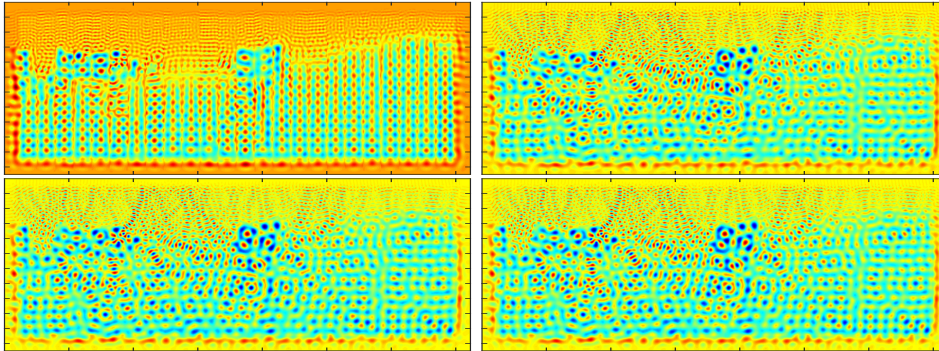
FIG. 5.2. *Two iterations of the preconditioner using a random source supported in the physical domain, from left to right and top to bottom: initial guess with only local solves; first iteration, second iteration, solution. The background model is given by the BP 2004 model [11] shown in Fig. 4.3.*

N	$\omega/2\pi$ [Hz]	$L \times L_c$	$s = \delta(\mathbf{x})$	$s = rand$
60×169	1.25	6×3	**3**	**3**
120×338	2.5	12×6	**4**	**4**
239×675	5.0	24×12	**4**	**5**
478×1349	10.0	48×24	**5**	**5**
955×2697	20.0	96×48	**5**	**6**

TABLE 5.2

Maximum number of iterations required to reduce the relative residual to 10^{-5}, using GMRES with the Gauss-Seidel preconditioner, for 50 realizations of a randomly located point source and a randomly generated global source. In this case, $f = \omega/2\pi \sim n$; the number of layers and the number of cells inside each layer grows as n; the number of points in the PML scales as $\log(N)$, and the sound speed is given by the BP 2004 model (see [11]).

larized system. In particular, the polarized system is solved iteratively using either GMRES or Bi-CGstab [95] preconditioned with either the Gauss-Seidel preconditioner (3.9) or the Jacobi preconditioner (3.10). Table 5.3 summarizes the typical behavior of the different combinations of preconditioners and Krylov solvers; it shows the number of iterations needed to converge when the polarized system (3.5) is solved for different frequencies and number of subdomains using the BP 2004 model.

We can observe that for this particular model, the number of iterations to converge depends weakly on the frequency and the number of subdomains when a horizontally layered decomposition is used. Moreover, when the system is preconditioned with P^{GS} the iterative solver converge twice as fast as when using P^{Jac} as a preconditioner, which can be loosely explained by the clustering of the eigenvalues of the preconditioned systems, as shown in Fig. 3.2. Table 5.3 shows that the number of iterations for Bi-CGstab are half the number of iterations for GMRES; however, Bi-CGstab needs two applications of the matrix and preconditioner per iteration, resulting in a comparable computational cost, albeit with a smaller memory foot-print for Bi-CGstab.

We point out that we used a zero initial guess for both GMRES and Bi-CGstab iterative solvers. Even though it is possible to use a random initial guess, it is not recommended given that the number of iterations will increase compared to a zero initial guess. Table 5.1 shows that the number of iterations to convergence using a random initial guess (generated using the `rand` command in MATLAB) retains the

N	$\omega/2\pi$ [Hz]	$L \times L_c$	GS_{gmres}	J_{gmres}	$GS_{bicgstab}$	$J_{bicgstab}$
120×338	2.50	9×3	**5**	**10**	**2.5**	**5**
239×675	5.0	18×6	**6**	**11**	**3**	**6**
478×1349	10.0	27×9	**6**	**12**	**3.5**	**6.5**
955×2697	20.0	36×12	**8**	**15**	**4**	**7**
1910×5394	40.0	45×15	**8**	**17**	**4.5**	**9.5**

TABLE 5.3

Number of iterations required to reduce the relative residual to 10^{-7}, using GMRES and BiCGstab preconditioned with the Gauss-Seidel and Jacobi preconditioners. $f = \omega/2\pi \sim n$; the number of points in the PML scales as $\log(N)$, and the sound speed is given by the BP 2004 model (see [11]).

same asymptotic behavior as before, albeit with larger constants.

5.3. Layered media. Although the asymptotic behavior of the sweeping type solvers seems to be weakly dependent of the frequency, the number of subdomains, the source, and the initial guess, this kind of methods can be very sensitive to the medium and to the orientation of the layered decomposition. One typical example of the former is a medium containing large resonant cavities, for which the number of iterations becomes highly frequency dependent. One example of the later is layered media with multi-scale structures at the wavelength level: if the domain decomposition is performed perpendicularly to the layers in the media (i.e., the layers in the decomposition are perpendicular to the layers present in the medium), the empirical behavior of the method remains the same; however, if the domain decomposition is performed parallel to the layers, there is a detriment on the convergence rate.

We showcase this behavior using two different parts of the Marmousi2 [71] model. We use the central part of the model, as in Fig. 5.3 (*left*), in which we observe several layers that span different length-scales, but without a preferred orientation; and the left-most part of the model, as in Fig. 5.3 (*right*), in which most of the layers are oriented horizontally. Table 5.4 shows the number of iterations to convergence for each portion of the model, using a decomposition with either horizontal or vertical layers. When the medium does not posses a preferred orientation, both decompositions performs roughly the same; however, when there is a preferred orientation the decomposition with layers perpendicular to the preferred orientation performs better. This difference in the performance is due to the increasing number of layers and frequency *simultaneously*; in fact, the number of iterations depends weakly on the frequency when the number of layers is constant as shown in Table 5.5.

We interpret the detriment of the convergence rate as follows: from [103] it is known that the performance of the method of polarized traces is roughly proportional to energy scattered back at each sweep compared to the energy transmitted, which is closely linked to the number of reflections of the wave-field when interacting with the medium. In particular, the interactions within each subdomain are treated seamlessly using a direct solver, and interactions across subdomains are treated iteratively. The method handles the propagating waves in a sequential fashion by sweeping the waves traveling across the subdomains, and treating the reflections in an ulterior sweep in the opposite direction. For example, if the medium is constant, then at each sweep most of the energy of the waves will be transmitted across subdomains, up to a small amount that is reflected given the imperfection of the absorbing boundary conditions. However, if the medium exhibits discontinuities in the wave speed, which will produce new reflexions, then energy will be scattered back. These new reflections need to be
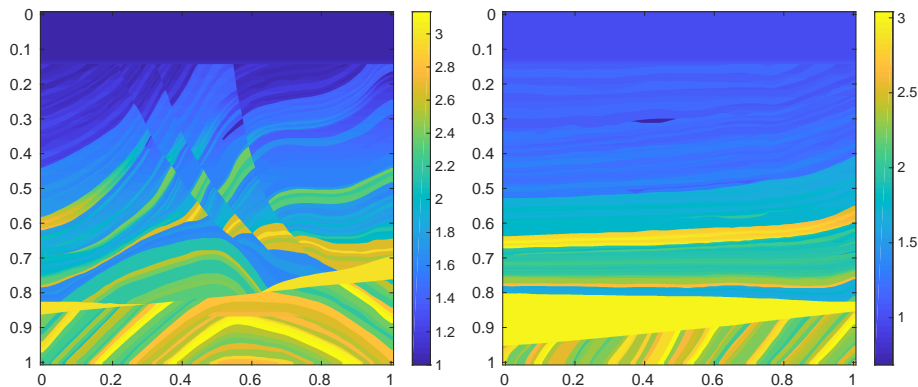
27

FIG. 5.3.   *Truncated Marmousi models: left: the central part without a preferred layered structure, right: left-most part, with a horizontally layered structure.*
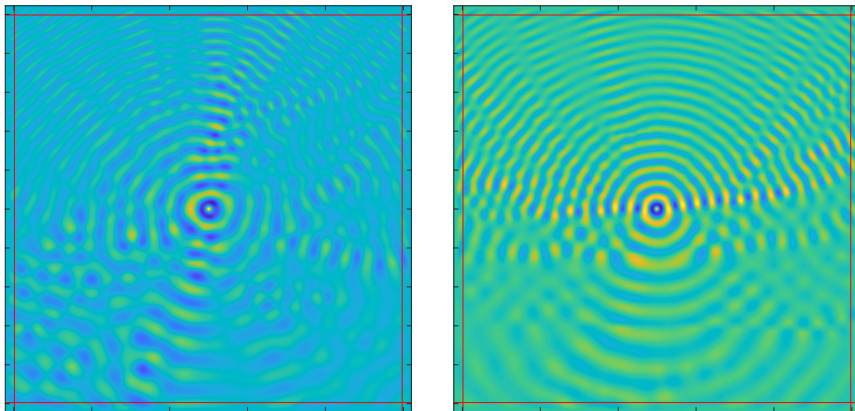

FIG. 5.4.   *Solutions to the Marmousi2 patches for $\omega/(2\pi) = 35$: left: solution to the Helmholtz equation for the central patch of the Marmousi2 model, right: solution to the Helmholtz equation for the left-most patch of the Marmousi2 model*

propagated in an ulterior sweep in the opposite direction, which will provoke new reflections that need to be propagated in a subsequent sweep and so on.

If the medium presents a handful of layers as in Fig. 5.5, and the frequency and the number of sub-domains increases we have that most of the subdomains will not present any interior reflections. Thus, during the sweeps the waves are transmitted seamlessly across most subdomains until they hit a subdomain containing a discontinuity, in which a reflection is created, which will need to be transmitted in an ulterior sweep. In this case, the number of reflections in the domain is mostly frequency independent. Therefore, provided that enough absorption is used for the transmission conditions between subdomains, the performance of the method is preserved as shown in Table 5.6.

It is well known that waves interact with structures that have a characteristic length of the same order than the wavelength. In the case of a layered medium with multi-scale structures, as the one presented in Fig. 5.3 (*right*), as the frequency increases, or the wavelength decreases, the waves will increasingly interact with increasingly thinner layers thus increasing the amount of energy scattered back.

If the frequency increases but the number of layers is constant, then the increased

28

N	$\omega/2\pi$	L	$Iter_{\text{cent,vert}}$	$Iter_{\text{cent,hor}}$	$Iter_{\text{lay,vert}}$	$Iter_{\text{lay,hor}}$
88×88	8	6	**4**	**4**	**3**	**4**
175×175	17	12	**4**	**5**	**3**	**6**
350×350	35	24	**5**	**6**	**4**	**7**
700×700	70	48	**6**	**7**	**4**	**7**
1400×1400	140	96	**7**	**8**	**5**	**12**
2800×2800	280	192	**8**	**9**	**7**	**18**

TABLE 5.4

Number of iterations required to reduce the relative residual to 10^{-5}, using GMRES preconditioned with the Gauss-Seidel preconditioner, in two different patches of the Marmousi2 model [71] using either a horizontal or vertical layered decomposition. The subscripts indicate in which medium the Helmholtz equation was solved: lay stands for the layered patch in Fig. 5.3 right; whereas, cent stands for the centered patch in Fig. 5.3 left. $f = \omega/2\pi \sim n$; the number of layers scales as $L \sim n$; and the number of points in the PML scales as $\log(N)$.

N	$\omega/2\pi$	L	$Iter_{\text{lay,vert}}$	$Iter_{\text{lay,hor}}$
88×88	8	6	**3**	**4**
175×175	17	6	**3**	**5**
350×350	35	6	**4**	**6**
700×700	70	6	**4**	**6**
1400×1400	140	6	**4**	**6**
2800×2800	280	6	**5**	**6**

TABLE 5.5

Number of iterations required to reduce the relative residual to 10^{-5}, using GMRES preconditioned with the Gauss-Seidel preconditioner, using the layered patch of the Marmousi2 model [71] as depicted in Fig. 5.3 (right), using either a horizontal or vertical layered decomposition. $f = \omega/2\pi \sim n$; the number of layers is fixed and equal to 6; and the number of points in the PML scales as $\log(N)$.

number of interaction due to the increment on the frequency are handled with a direct method within each subdomain, the only interactions handled iteratively are the transmission and reflection between subdomains, which remains almost constant as the frequency increases, as depicted by Table 5.5.

However, if the number of domains increases as the frequency increases, then the orientation of the domain decomposition is crucial. If the orientation is perpendicular to the layered media, then, as the frequency increases, the energy scattered from the micro structures is efficiently handled via a direct method within each subdomain. The interaction between subdomains will be composed of mostly transmitted waves propagating within each of the layers present in the medium. These interactions are efficiently handled by the sweeps. In such cases the method retains its performance as shown in Table 5.4. However, if the domain decomposition aligned to the layered media, then the method needs to properly handle all the internal reflection between the increasingly numbers of subdomains in a sequential fashion, thus resulting in an increasingly number of iterations to convergence as shown in Table 5.4.

Finally, the performance of the method is not only function of the number of layers, it is function of the strength of the reflections, which is directly linked to the contrast between layers. For example in a layered medium with strong contrast, the amount of energy scattered back will increase, meaning that the sweeps needs to take in account more reflections, as shown in Table 5.6.

We point out that in these numerical experiments we used $L \sim n$, which is a much more astringent conditions that the one used in the nested version of the
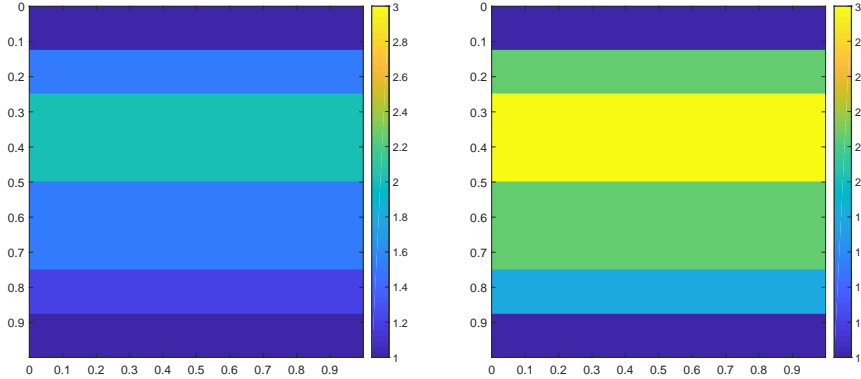
FIG. 5.5. *Layered media used for the numerical experiments with different contrast, the left medium has a maximum contrast of 2.0 in the wave speed, and the right medium has a maximum contrast of 3.0.*

N	$\omega/2\pi$	L	$Iter_{2.0}$	$Iter_{3.0}$
60×60	6	4	**2**	**2**
120×120	12	8	**2**	**2**
240×240	24	16	**2**	**3**
480×480	48	32	**3**	**4**
960×960	96	64	**3**	**5**
1920×1920	192	128	**5**	**7**

TABLE 5.6

Number of iterations required to reduce the relative residual to 10^{-5}, using GMRES preconditioned with the Gauss-Seidel preconditioner, for the layered models in Fig. 5.5, using a vertical domain decomposition, for two different contrasts. $f = \omega/2\pi \sim n$; the number of layers scales as $L \sim n$; and the number of points in the PML scales as $\log(N)$.

algorithm, in which $L \sim L_c \sim \sqrt{P}$, where $P = \mathcal{O}(N^{1/5})$. In order to keep an online complexity sublinear, the number of layers and cells in the decomposition grows slowly with respect to the frequency. This fact coupled with the very efficient transmission conditions between cells, greatly attenuates the grow in the number of iterations for the nested solver within a layer, even in the case of multi-scale layers with a preferred orientation.

5.4. Fast methods. Table 5.3 depicts the efficiency of the preconditioner measured in the number of iterations for convergence; however, in order to obtain sublinear runtimes we need to compress the integral kernels. As noted in [103] the scaling of the number of degrees of freedom with the frequency is critical to obtain the correct asymptotic compression rate. If the scaling is too aggressive, as in Table 5.3, the pollution error will be overwhelming and the compression rate of the integral operator will suffer. In order to account for the pollution error, in Table 5.7 we use the scaling $n \sim \omega^2$, which is known to be quasi-optimal for finite-elements and finite differences (even though it is widely believed that $n \sim \omega^{3/2}$ is enough, cf. [5])

Table 5.7 shows the sublinear[18] $\mathcal{O}(P^{1-\alpha}N^{\alpha} + P\log(N)^{\alpha})$ scaling of the runtime

---

[18]The same scaling hold for other typical geophysical benchmarks such as Marmousi2 [71], in which convergence is achieved in 4-6 iterations.

N	$\omega/2\pi$ [Hz]	6×2	24×8	42×14	60×20
120×338	2.50	**(4)** 0.42	**(4)** 8.30	**(4)** 24.8	**(4)** 51.7
239×675	3.56	**(4)** 0.74	**(5)** 9.15	**(5)** 26.1	**(5)** 52.8
478×1349	5.11	**(4)** 1.52	**(5)** 11.6	**(5)** 30.8	**(5)** 59.9
955×2697	7.25	**(5)** 3.32	**(5)** 17.9	**(6)** 38.5	**(6)** 68.8
1910×5394	10.3	**(5)** 6.79	**(6)** 29.6	**(6)** 58.7	**(6)** 98.3

Table 5.7

Number of GMRES iterations (bold) required to reduce the relative residual to 10^{-5}, along with average execution time (in seconds) of one GMRES iteration using the compressed direct method, for different N and $P = L \times L_c$. The frequency is scaled such that $f = \omega/2\pi \sim \sqrt{n}$, the number of points in the PML scales as $\log(N)$, and the sound speed is given by the BP 2004 model (see [11]).
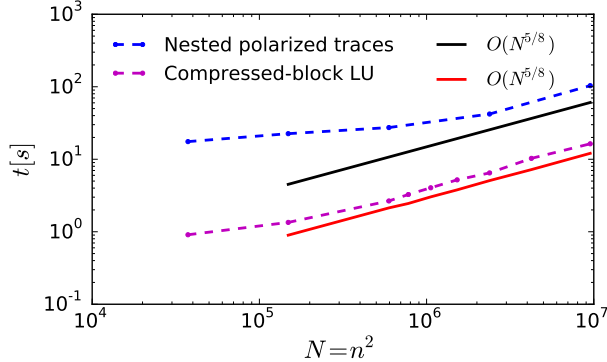

Fig. 5.6. *Runtime for one GMRES iteration using the two different nested solves, for $L = 9$ and $L_c = 3$, $\omega \sim \sqrt{n}$, in which the maximum ϵ-rank in the adaptive PLR compression (see Section 5.1 in [103] for further details) scales as $max_{rank} \sim \sqrt{\omega}$ and $\epsilon = 10^{-8}$. Moreover, for the nested polarized traces, the accuracy for the GMRES inner solve is fixed to 10^{-6}.*

of one GMRES iteration for $\alpha = 5/8$, as shown by Fig. 5.6. Once again we can observe that the number of iterations to converge depends weakly on the frequency and the number of subdomains. From Fig. 5.6 depicts the efficient compression of the discrete integral operators, we can observe that both methods (nested polarized traces and compressed-block LU) have the same asymptotic runtime, but with different constants[19], the same scaling holds for different numbers of cells and layers.

6. Conclusion. We presented an extension to the method of polarized traces introduced in [103], with improved asymptotic runtimes in a distributed memory environment. The method has sublinear runtime even in the presence of rough media of geophysical interest. Moreover, its performance is completely agnostic to the source.

The method can be embedded efficiently within algorithms that require to solve systems in which the medium is locally updated in an inversion loop. The method only needs to be locally modified in order to solve the updated system, thus reducing the overall computational effort. This algorithm is of special interest in the context of time-lapse full-waveform inversion, continuum reservoir monitoring, and local inversion.

We point out that this approach can be further parallelized using distributed linear algebra libraries. Moreover, it is possible to solve multiple right-hand sides simultaneously, without an asymptotic penalty. This can be achieved by pipelining

---

[19]We point out that some gains can be made by using different compressed operators. One can use one compressed operator with high accuracy to apply $\underline{\mathbf{M}}$, an operation that is easily parallelizable, and another with low accuracy to apply the preconditioner that represents the sequential bottleneck.

the sweeps, i.e., performing additional sweeps before the first one has finished, in order to maintain a constant load among all the nodes.

Appendix A. Discretization using finite differences.

In order to impose the absorbing boundary conditions, we extend the rectangular domain $\Omega = (0, L_x) \times (0, L_z)$ to $\Omega^{\text{ext}} = (-\delta_{\text{pml}}, L_x + \delta_{\text{pml}}) \times (-\delta_{\text{pml}}, L_z + \delta_{\text{pml}})$. The Helmholtz operator in (2.1) then takes the form

$$\mathcal{H} = -\partial_{xx} - \partial_{zz} - m\omega^2, \qquad \text{in } \Omega^{\text{ext}}, \tag{A.1}$$

where m is an extension[20] of the squared slowness. The differential operators are redefined following

$$\partial_x \to \alpha_x(\mathbf{x})\partial_x, \qquad \partial_z \to \alpha_z(\mathbf{x})\partial_z, \tag{A.2}$$

where

$$\alpha_x(\mathbf{x}) = \frac{1}{1 + i\frac{\sigma_x(\mathbf{x})}{\omega}}, \qquad \alpha_z(\mathbf{x}) = \frac{1}{1 + i\frac{\sigma_z(\mathbf{x})}{\omega}}. \tag{A.3}$$

Moreover, $\sigma_x(\mathbf{x})$ is defined as

$$\sigma_x(\mathbf{x}) = \begin{cases} \frac{C}{\delta_{\text{pml}}}\left(\frac{x}{\delta_{\text{pml}}}\right)^2, & \text{if } x \in (-\delta_{\text{pml}}, 0), \\ 0, & \text{if } x \in [0, L_x], \\ \frac{C}{\delta_{\text{pml}}}\left(\frac{x - L_x}{\delta_{\text{pml}}}\right)^2, & \text{if } x \in (L_x, L_x + \delta_{\text{pml}}), \end{cases} \tag{A.4}$$

and similarly for $\sigma_z(\mathbf{x})$ [21]. In general, δ_{pml} goes from a couple of wavelengths in a uniform medium, to a large number independent of ω in a highly heterogeneous medium; and C is chosen to provide enough absorption.

With this notation we rewrite (2.1) as

$$\mathcal{H}u = f, \qquad \text{in } \Omega^{\text{ext}}, \tag{A.5}$$

with homogeneous Dirichlet boundary conditions (f is the zero extended version of f to Ω^{ext}).

We discretize Ω as an equispaced regular grid of stepsize h, and of dimensions $n_x \times n_z$. For the extended domain Ω^{ext}, we extend this grid by $n_{\text{pml}} = \delta_{\text{pml}}/h$ points in each direction, obtaining a grid of size $(2n_{\text{pml}} + n_x) \times (2n_{\text{pml}} + n_z)$. Define $\mathbf{x}_{p,q} = (x_p, z_q) = (ph, qh)$.

We use the 5-point stencil Laplacian to discretize (A.5). For the interior points $\mathbf{x}_{i,j} \in \Omega$, we have

$$(\mathbf{H}\mathbf{u})_{p,q} = -\frac{1}{h^2}\left(\mathbf{u}_{p-1,q} - 2\mathbf{u}_{p,q} + \mathbf{u}_{p+1,q}\right) - \frac{1}{h^2}\left(\mathbf{u}_{p,q-1} - 2\mathbf{u}_{p,q} + \mathbf{u}_{p,q+1}\right) - \omega^2 m(\mathbf{x}_{p,q}).$$
$$\tag{A.6}$$

In the PML, we discretize $\alpha_x \partial_x(\alpha_x \partial_x u)$ as

$$\alpha_x(\mathbf{x}_{p,q})\frac{\alpha(\mathbf{x}_{p+1/2,q})(\mathbf{u}_{p+1,q} - \mathbf{u}_{p,q}) - \alpha_x(\mathbf{x}_{p-1/2,q})(\mathbf{u}_{p,q} - \mathbf{u}_{p-1,q})}{h^2}, \tag{A.7}$$

---

[20] We assume that $m(x)$ is given to us in Ω^{ext}.

[21] In practice, δ_{pml} and C can be seen as parameters to be tuned for accuracy versus efficiency.

and analogously for $\alpha_z \partial_z(\alpha_z \partial_z u)$. Although we use a second order finite difference stencil, the method can be easily extended to higher order finite differences. In such cases, the number of traces needed in the SIE reduction will increase.

Appendix B. Discretization using Q1 finite elements. The symmetric formulation of the Helmholtz equation takes the form

$$- \left(\nabla \cdot \Lambda \nabla + \frac{\omega^2 m(\mathbf{x})}{\alpha_x(\mathbf{x})\alpha_z(\mathbf{x})} \right) u(\mathbf{x}) = \frac{f(\mathbf{x})}{\alpha_x(\mathbf{x})\alpha_z(\mathbf{x})}, \tag{B.1}$$

where

$$\Lambda(\mathbf{x}) = \left[\begin{array}{cc} s_x(\mathbf{x}) & 0 \\ 0 & s_z(\mathbf{x}) \end{array} \right], \tag{B.2}$$

$s_x = \alpha_x/\alpha_z$, $s_z = \alpha_z/\alpha_x$, where α_x and α_z are defined in (A.3).

In the case of a medium with sharp interfaces, finite differences approximations give inaccurate results due to the lack of differentiability of the velocity profile. In such cases, sophisticated quadratures and adaptive meshes have to be implemented to properly approximate the finite difference operator [100, 3]. We opted for a low order Q1 finite element discretization, with an adaptive quadrature rule at the discontinuities.

(B.1) is discretized using Q1 elements, leading to a discretized matrix

$$\mathbf{H} = \mathbf{S} - \mathbf{M}, \tag{B.3}$$

where the stiffness matrix $\mathbf{S}$ is computed using a Gauss quadrature. On the other hand the mass matrix, $\mathbf{M}$, is computed using a quadrature adapted to each element depending on the local smoothness of the velocity profile:

- if the medium is locally smooth, a fixed Gauss quadrature is used to approximate the integral over the square;
- if the medium is discontinuous, an adaptive trapezoidal rule is used, until a preset accuracy is achieved.

To discriminate whether the medium is discontinuous, the velocity is sampled at the Gauss-points, and the ratio between maximum and minimum velocity is computed. If the ratio is smaller than a fixed threshold, the medium is considered smooth; otherwise it is considered discontinuous.

Using a nodal basis we can write the system to solve as

$$\mathbf{H}\mathbf{u} = \mathbf{f}, \tag{B.4}$$

where $\mathbf{u}$ is the point-wise value of the solution at the corners of the mesh and $\mathbf{f}$ is the projection of f onto the Q1 elements, using a high-order quadrature rule.

Appendix C. Green's representation formula.

We present a generalization of the domain decomposition framework developed in [103] to Q1 regular finite elements.

We start by providing the algebraic formula for the discrete Green's representation formula. We propose a technique to derive such formulas without the time consuming computations performed in the Appendix of [103]. We point out that there are clear parallels between this derivation and the reduction to an interface problem using interior Schur complements. However, for the interface system based on Schur complements, we could not define the polarizing conditions that would allows us to

construct an efficient preconditioner, forcing us to solve it using a non-scalable direct solver.

In the sequel, we perform extensive manipulations on the matrix $\mathbf{H}$, thus we introduce some notation to help the reader follow the computations. Following that ordering define in Section 2.2 we write $\mathbf{H}$ (and $\mathbf{H}^\ell$) as a block matrix in the form

$$
\mathbf{H} = \begin{bmatrix}
\mathbf{H}_{1,1} & \mathbf{H}_{1,2} & & & \\
\mathbf{H}_{2,1} & \mathbf{H}_{2,2} & \mathbf{H}_{2,3} & & \\
& \ddots & \ddots & & \ddots & \\
& & \ddots & & \ddots & \mathbf{H}_{n_z-1,n_z} \\
& & & \mathbf{H}_{n_z,n_z-1} & \mathbf{H}_{n_z,n_z}
\end{bmatrix},
\tag{C.1}
$$

in which each block correspond to a fixed z.

We want to derive the algebraic form of the Green's representation formula. From Theorem 1 in [103] we know that using the Green's representation formula locally in a subdomain would produce a discontinuous solution, such that the exact solution is recovered inside the domain, and it is zero outside it. The rationale behind the formalism presented in this section is to find the form of the forcing terms necessary to force the discontinuity of the local representation[22].

An easy manner to deduce the Green's representation formula is to let

$$
\mathbf{v}^\ell = \mathbf{u}\chi_{\Omega^\ell},
\tag{C.2}
$$

which is discontinuous, and apply the local differential operator to $\mathbf{v}^\ell$. Finding the discrete Green's representation formula can be re-cast as finding the expression of a system of the form

$$
\mathbf{H}^\ell \mathbf{v}^\ell = \mathbf{f}^\ell + \mathcal{F}^\ell(\mathbf{u}),
\tag{C.3}
$$

such that its solution $\mathbf{v}^\ell$ satisfies $\mathbf{v}^\ell = \mathbf{u}\chi_{\Omega^\ell}$, and $\mathcal{F}$ depends on the global wavefield $\mathbf{u}$. In (C.3) we suppose that $\mathbf{f}^\ell = \mathbf{f}\chi^\ell$ and that $\mathbf{H}$ and $\mathbf{H}^\ell$ coincide exactly inside the layer. Within this context the problem of finding the formula for the Green's representation formula can be reduced to finding the expression of $\mathcal{F}^\ell(\mathbf{u})$ such that $\mathbf{v}^\ell$ satisfies Eq C.2.

For ℓ fixed we can obtain the expression of $\mathcal{F}^\ell$ by evaluating (C.3) and imposing that $\mathbf{v}^\ell = \mathbf{u}\chi_{\Omega^\ell}$. In particular, we need to evaluate (C.3) at the interior of the slab, at its boundaries and at the exterior.

At the interior of the slab $\mathcal{F}^\ell(\mathbf{u})$ is zero, because $\mathbf{v}^\ell$ satisfies $\mathbf{H}^\ell \mathbf{v}^\ell = \mathbf{H}\mathbf{u} = \mathbf{f} = \mathbf{f}^\ell$.

At the boundaries, the situation is slightly more complex. If we evaluate (C.3) at $k = 1$, we have that

$$
\mathbf{H}_{1,1}^\ell \mathbf{v}_1^\ell + \mathbf{H}_{1,2}^\ell \mathbf{v}_2^\ell = \mathbf{f}_1^\ell + \mathcal{F}_1(\mathbf{u}).
\tag{C.4}
$$

Moreover, evaluating $\mathbf{H}\mathbf{u} = \mathbf{f}$ at the same index yields

$$
\mathbf{H}_{1,0}\mathbf{u}_0 + \mathbf{H}_{1,1}\mathbf{u}_1 + \mathbf{H}_{1,2}\mathbf{u}_2 = \mathbf{f}_1^\ell.
\tag{C.5}
$$

By imposing that $\mathbf{v}^\ell = \mathbf{u}\chi_{\Omega^\ell}$ and subtracting (C.4) and (C.5), we have that

$$
\mathcal{F}_1^\ell(\mathbf{u}) = -\mathbf{H}_{1,0}\mathbf{u}_0 = -\mathbf{H}_{1,0}^\ell \mathbf{u}_0.
\tag{C.6}
$$

---

[22]The technique to compute the Green's representation formula, and therefore the transmission operators in form of an incomplete Green's integral, was first mentioned, to the authors knowledge, in Appendix 3B in [101] . More recently, an analogous formulation was used in [89] (see Eqs. (11), (12) and (13).

We can observe that the role of $\mathcal{F}^\ell$ is to complete (C.3) at the boundary with exterior data, such that $\mathbf{v}^\ell$ satisfies the same equation that $\mathbf{u}$ inside the whole layer and not only in the interior.

Analogously (C.3) can be evaluated at $k = 0$ obtaining

$$\mathbf{H}^\ell_{0,1}\mathbf{v}^\ell_1 = \mathcal{F}^\ell_0(\mathbf{u}), \tag{C.7}$$

and imposing that $\mathbf{v}^\ell = \mathbf{u}\chi_{\Omega^\ell}$ we obtain that

$$\mathcal{F}^\ell_0(\mathbf{u}) = \mathbf{H}_{0,1}\mathbf{u}_1 = \mathbf{H}^\ell_{0,1}\mathbf{u}_1. \tag{C.8}$$

Finally, for $k < 0$, the same argument leads to

$$\mathcal{F}^\ell_k(\mathbf{u}) = 0. \tag{C.9}$$

We can easily generalize this argument for the other side of a layer obtaining a generic formula for $\mathcal{F}^\ell$

$$\mathcal{F}^\ell(\mathbf{u}) = -\delta_{n^\ell}\mathbf{H}^\ell_{n^\ell,n^\ell+1}\mathbf{u}_{n^\ell+1} + \delta_{n^\ell+1}\mathbf{H}^\ell_{n^\ell+1,n^\ell}\mathbf{u}_{n^\ell} - \delta_1\mathbf{H}^\ell_{1,0}\mathbf{u}_0 + \delta_0\mathbf{H}^\ell_{0,1}\mathbf{u}_1,$$

which can be substituted in (C.3), leading to

$$\begin{aligned}\mathbf{H}^\ell\mathbf{v}^\ell = -\,&\delta_{n^\ell}\mathbf{H}^\ell_{n^\ell,n^\ell+1}\mathbf{u}_{n^\ell+1} + \delta_{n^\ell+1}\mathbf{H}^\ell_{n^\ell+1,n^\ell}\mathbf{u}_{n^\ell} \\ -\,&\delta_1\mathbf{H}^\ell_{1,0}\mathbf{u}_0 + \delta_0\mathbf{H}^\ell_{0,1}\mathbf{u}_1 + \mathbf{f}^\ell.\end{aligned} \tag{C.10}$$

In addition, (C.10) can be transformed into the discrete expression of the Green's representation formula by applying the inverse of $\mathbf{H}^\ell$, $\mathbf{G}^\ell$. We can then reformulate the Green's integral in Def. 2.2 in the form

$$\mathcal{G}^{\downarrow,\ell}_j(\mathbf{v}_0,\mathbf{v}_1) = h\begin{bmatrix}\mathbf{G}^\ell(z_j,z_1) & \mathbf{G}^\ell(z_j,z_0)\end{bmatrix}\begin{pmatrix}-\mathbf{H}^\ell_{1,0}\mathbf{v}_0 \\ \mathbf{H}^\ell_{0,1}\mathbf{v}_1\end{pmatrix}, \tag{C.11}$$

$$\mathcal{G}^{\uparrow,\ell}_j(\mathbf{v}_{n^\ell},\mathbf{v}_{n^\ell+1}) = h\begin{bmatrix}\mathbf{G}^\ell(z_j,z_{n^\ell+1}) & \mathbf{G}^\ell(z_j,z_{n^\ell})\end{bmatrix}\begin{pmatrix}\mathbf{H}^\ell_{n^\ell+1,n^\ell}\mathbf{v}_{n^\ell} \\ -\mathbf{H}^\ell_{n^\ell,n^\ell+1}\mathbf{v}_{n^\ell+1}\end{pmatrix} \tag{C.12}$$

Finally, we can redefine $\mathbf{G}^\ell(z_j,z_k)$ for $k = 0,1,n^\ell,n^\ell+1$, such that they absorb all the extra factors. In particular, we redefine:

$$\begin{aligned}\mathbf{G}^\ell_{1,0} &= -\,\delta_1\left((\mathbf{H}^\ell)^{-1}\delta_0\mathbf{H}^\ell_{1,0}\right), & \mathbf{G}^\ell_{1,1} &= -\delta_1\left((\mathbf{H}^\ell)^{-1}\delta_1\mathbf{H}^\ell_{0,1}\right), & \text{(C.13)}\\ \mathbf{G}^\ell_{n,n} &= -\,\delta_n\left((\mathbf{H}^\ell)^{-1}\delta_{n+1}\mathbf{H}^\ell_{n+1,n}\right), & \mathbf{G}^\ell_{n,n+1} &= -\delta_n\left((\mathbf{H})^{-1}\delta_n\mathbf{H}_{n,n+1}\right). & \text{(C.14)}\end{aligned}$$

This redefinition allows us to seamlessly use all the machinery introduced in [103] to define the SIE and its preconditioner.

REMARK 1. *As an example, in the case of the unsymmetric finite difference discretization, the upper and lower diagonal blocks of $\mathbf{H}$ are diagonal matrices rescaled by $-1/h^2$. Then the formula presented here reduces exactly to the formulas computed by summation by parts in Appendix of [103].*

REFERENCES

[1] S. Ambikasaran, C. Borges, L.-M. Imbert-Gerard, and L. Greengard. Fast, adaptive, high order accurate discretization of the Lippmann-Schwinger equation in two dimension. *ArXiv e-prints*, [math.NA] 1505.07157, 2015.

[2] P. R. Amestoy, I. S. Duff, J.-Y. L'Excellent, and J. Koster. A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM Journal on Matrix Analysis and Applications*, 23(1):15–41, 2001.

[3] I. Babuska, F. Ihlenburg, E. T. Paik, and S. A. Sauter. A generalized finite element method for solving the Helmholtz equation in two dimensions with minimal pollution. *Computer Methods in Applied Mechanics and Engineering*, 128(3-4):325–359, 1995.

[4] I. Babuska and J. M. Melenk. The partition of unity method. *International Journal for Numerical Methods in Engineering*, 40(4):727–758, 1997.

[5] A. Bayliss, C. I. Goldstein, and Turkel E. On accuracy conditions for the numerical computation of waves. *Journal of Computational Physics*, 59(3):396 – 404, 1985.

[6] M. Bebendorf. *Hierarchical Matrices: A Means to Efficiently Solve Elliptic Boundary Value Problems*, volume 63 of *Lecture Notes in Computational Science and Engineering (LNCSE)*. Springer-Verlag, 2008. ISBN 978-3-540-77146-3.

[7] M. Bebendorf and W. Hackbusch. Existence of $\mathcal{H}$-matrix approximants to the inverse fe-matrix of elliptic operators with l^∞-coefficients. *Numerische Mathematik*, 95(1):1–28, 2002.

[8] R. Bélanger-Rioux and L. Demanet. Compressed absorbing boundary conditions via matrix probing. *ArXiv e-prints*, [math.NA] 1401.4421, 2014.

[9] J.-P. Bérenger. A perfectly matched layer for the absorption of electromagnetic waves. *Journal of Computational Physics*, 114(2):185–200, 1994.

[10] A. Bermúdez, L. Hervella-Nieto, A. Prieto, and R. Rodríguez. Perfectly matched layers for time-harmonic second order elliptic problems. *Archives of Computational Methods in Engineering*, 17(1):77–107, 2010.

[11] F. Billette and S. Brandsberg-Dahl. *The 2004 BP velocity benchmark.* EAGE, 2005.

[12] S. Borm. Directional H2-matrix compression for high-frequency problems. *ArXiv e-prints*, [math.NA] 1510.07087, 2015.

[13] Y. Boubendir. An analysis of the BEM-FEM non-overlapping domain decomposition method for a scattering problem. *Journal of Computational and Applied Mathematics*, 204(2):282 – 291, 2007. Special Issue: The Seventh International Conference on Mathematical and Numerical Aspects of Waves (WAVES05).

[14] Y. Boubendir, X. Antoine, and C. Geuzaine. A quasi-optimal non-overlapping domain decomposition algorithm for the Helmholtz equation. *Journal of Computational Physics*, 231(2):262 – 280, 2012.

[15] A. Brandt and I. Livshits. Wave-ray multigrid method for standing wave equations. *Electronic Transactions on Numerical Analysis*, 6:162–181, 1997.

[16] H. Calandra, S. Gratton, X. Pinel, and X. Vasseur. An improved two-grid preconditioner for the solution of three-dimensional Helmholtz problems in heterogeneous media. *Numerical Linear Algebra with Applications*, 20(4):663–688, 2013.

[17] O. Cessenat and B. Després. Using plane waves as base functions for solving time harmonic equations with the ultra weak variational formulation. *Journal of Computational Acoustics*, 11(02):227–238, 2003.

[18] T. F. Chan and T. P. Mathew. Domain decomposition algorithms. *Acta Numerica*, 3:61–143, 1 1994.

[19] S. N. Chandler-Wilde, I. G. Graham, S. Langdon, and E. A. Spence. Numerical-asymptotic boundary integral methods in high-frequency acoustic scattering. *Acta Numerica*, 21:89–305, 5 2012.

[20] Z. Chen, M. J. Gander, and H. Zhang. *Domain Decomposition Methods in Science and Engineering XXII*, chapter On the Relation Between Optimized Schwarz Methods and Source Transfer, pages 217–225. Springer International Publishing, Cham, 2016.

[21] Z. Chen and X. Xiang. A source transfer domain decomposition method for Helmholtz equations in unbounded domain. *SIAM Journal on Numerical Analysis*, 51(4):2331–2356, 2013.

[22] Z. Chen and X. Xiang. A source transfer domain decomposition method for Helmholtz equations in unbounded domain part II: Extensions. *Numerical Mathematics: Theory, Methods and Applications*, 6:538–555, 2013.

[23] J. Chiu and L. Demanet. Matrix probing and its conditioning. *SIAM Journal on Numerical Analysis*, 50(1):171–193, 2012.

[24] X. Claeys and R. Hiptmair. Multi-trace boundary integral formulation for acoustic scattering by composite structures. *Communications on Pure and Applied Mathematics*, 66(8):1163–1201, 2013.

[25] F. Collino, S. Ghanemi, and P. Joly. Domain decomposition method for harmonic wave propagation: a general presentation. *Computer Methods in Applied Mechanics and En-*

gineering, 184(24):171 – 211, 2000.

[26] S. Cools and W. Vanroose. Local Fourier analysis of the complex shifted Laplacian preconditioner for Helmholtz problems. *Numerical Linear Algebra with Applications*, 20(4):575–597, 2013.

[27] T. A. Davis. Algorithm 832: UMFPACK v4.3—an unsymmetric-pattern multifrontal method. *ACM Transactions on Mathematical Software*, 30(2):196–199, June 2004.

[28] A. de La Bourdonnaye, C. Farhat, A. Macedo, F. Magoules, and F.-X. Roux. A non-overlapping domain decomposition method for the exterior Helmholtz problem. *Contemporary Mathematics*, 218:42–66, 1998.

[29] B. Després. Décomposition de domaine et problème de Helmholtz. *Comptes rendus de l'Académie des sciences. Série 1, Mathématique*, 311:313–316, 1990.

[30] Y. Du and H. Wu. An improved pure source transfer domain decomposition method for Helmholtz equations in unbounded domain. *ArXiv e-prints*, 2015.

[31] I. S. Duff and J. K. Reid. The multifrontal solution of indefinite sparse symmetric linear. *ACM Trans. Math. Softw.*, 9(3):302–325, September 1983.

[32] B. Engquist and L. Ying. Sweeping preconditioner for the Helmholtz equation: Hierarchical matrix representation. *Communications on Pure and Applied Mathematics*, 64(5):697–735, 2011.

[33] B. Engquist and L. Ying. Sweeping preconditioner for the Helmholtz equation: moving perfectly matched layers. *Multiscale Modeling & Simulation*, 9(2):686–710, 2011.

[34] B. Engquist and H.-K. Zhao. Absorbing boundary conditions for domain decomposition. *Applied Numerical Mathematics*, 27(4):341 – 365, 1998. Special Issue on Absorbing Boundary Conditions.

[35] B. Engquist and H.-K. Zhao. Approximate separability of Green's function for high frequency Helmholtz equations, March 2014.

[36] Y. A. Erlangga. Advances in iterative methods and preconditioners for the Helmholtz equation. *Archives of Computational Methods in Engineering*, 15(1):37–66, 2008.

[37] Y. A. Erlangga, C. W. Oosterlee, and C. Vuik. A novel multigrid based preconditioner for heterogeneous Helmholtz problems. *SIAM Journal on Scientific Computing*, 27(4):1471–1492, 2006.

[38] O. G. Ernst and M. J. Gander. Why it is difficult to solve Helmholtz problems with classical iterative methods. In Ivan G. Graham, Thomas Y. Hou, Omar Lakkis, and Robert Scheichl, editors, *Numerical Analysis of Multiscale Problems*, volume 83 of *Lecture Notes in Computational Science and Engineering*, pages 325–363. Springer Berlin Heidelberg, 2012.

[39] C. Farhat, I. Harari, and L. P. Franca. The discontinuous enrichment method. *Computer Methods in Applied Mechanics and Engineering*, 190(48):6455–6479, 2001.

[40] M. J. Gander. Optimized schwarz methods. *SIAM Journal on Numerical Analysis*, 44(2):699–731, 2006.

[41] M. J. Gander. On the limitations of sweeping type preconditioners for wave propagation. *Mini-Workshop on Fast Solvers for Highly Oscillatory Problems, Oberwolfach Report*, 50, 2016.

[42] M. J. Gander and F. Kwok. Optimal interface conditions for an arbitrary decomposition into subdomains. In Yunqing Huang, Ralf Kornhuber, Olof Widlund, and Jinchao Xu, editors, *Domain Decomposition Methods in Science and Engineering XIX*, volume 78 of *Lecture Notes in Computational Science and Engineering*, pages 101–108. Springer Berlin Heidelberg, 2011.

[43] M. J. Gander, F. Magoulès, and F. Nataf. Optimized Schwarz methods without overlap for the Helmholtz equation. *SIAM Journal on Scientific Computing*, 24(1):38–60, 2002.

[44] M. J. Gander and F. Nataf. AILU for Helmholtz problems: A new preconditioner based on the analytic parabolic factorization. *Journal of Computational Acoustics*, 09(04):1499–1506, 2001.

[45] M. J. Gander and Y. Xu. *Domain Decomposition Methods in Science and Engineering XXII*, chapter Optimized Schwarz Method with Two-Sided Transmission Conditions in an Unsymmetric Domain Decomposition, pages 631–639. Springer International Publishing, Cham, 2016.

[46] M. J. Gander and H. Zhang. Domain decomposition methods for the Helmholtz equation: A numerical investigation. In Randolph Bank, Michael Holst, Olof Widlund, and Jinchao Xu, editors, *Domain Decomposition Methods in Science and Engineering XX*, volume 91 of *Lecture Notes in Computational Science and Engineering*, pages 215–222. Springer Berlin Heidelberg, 2013.

[47] M. J. Gander and H. Zhang. Algorithmic perspective of PML transmission conditions for

domain decomposition methods. In *Antenna Measurements Applications (CAMA), 2014 IEEE Conference on*, pages 1–4, Nov 2014.

[48] M. J. Gander and H. Zhang. *Domain Decomposition Methods in Science and Engineering XXI*, chapter Optimized Schwarz Methods with Overlap for the Helmholtz Equation, pages 207–215. Springer International Publishing, Cham, 2014.

[49] M.J. Gander, I.G. Graham, and E.A. Spence. Applying GMRES to the Helmholtz equation with shifted Laplacian preconditioning: what is the largest shift for which wavenumber-independent convergence is guaranteed? *Numerische Mathematik*, pages 1–48, 2015.

[50] A. George. Nested dissection of a regular finite element mesh. *SIAM Journal on Numerical Analysis*, 10:345–363, 1973.

[51] S. Ghanemi. A domain decomposition method for Helmholtz scattering problems. In *Ninth International Conference on Domain Decomposition Methods*, pages 105–112, 1998.

[52] A. Gillman, A.H. Barnett, and P.G. Martinsson. A spectrally accurate direct solution technique for frequency-domain scattering problems with variable media. *BIT Numerical Mathematics*, pages 1–30, 2014.

[53] C. J. Gittelson, R. Hiptmair, and I. Perugia. Plane wave discontinuous Galerkin methods: Analysis of the h-version. *ESAIM: Mathematical Modelling and Numerical Analysis*, 43:297–331, 3 2009.

[54] N. Gmati and B. Philippe. *Comments on the GMRES Convergence for Preconditioned Systems*, pages 40–51. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[55] I. G. Graham, E. A. Spence, and E. Vainikko. Domain decomposition preconditioning for high-frequency Helmholtz problems using absorption. *ArXiv e-prints*, 2015.

[56] E. Haber and S. MacLachlan. A fast method for the solution of the Helmholtz equation. *Journal of Computational Physics*, 230(12):4403–4418, 2011.

[57] R. Hiptmair and C. Jerez-Hanckes. Multiple traces boundary integral formulation for Helmholtz transmission problems. *Advances in Computational Mathematics*, 37(1):39–91, 2012.

[58] R. Hiptmair, A. Moiola, and I. Perugia. Plane wave discontinuous Galerkin methods for the 2d Helmholtz equation: Analysis of the p-version. *SIAM Journal on Numerical Analysis*, 49(1):264–284, 2011.

[59] R. Hiptmair, A. Moiola, and I. Perugia. A survey of Trefftz methods for the Helmholtz equation. *ArXiv e-prints*, 2015.

[60] M. Gu J. Xia, S. Chandrasekaran and X. S. Li. Superfast multifrontal method for large structured linear systems of equations. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1382–1411, 2010.

[61] S. Johnson. Notes on perfectly matched layers (PMLs), March 2010.

[62] R. Kress. Linear integral equations. In *Applied Mathematical Sciences*. Springer, 1999.

[63] W. Leng. A fast propagation method for the Helmholtz equation. *ArXiv e-prints*, 2015.

[64] W. Leng and L. Ju. An overlapping domain decomposition preconditioner for the Helmholtz equation. *ArXiv e-prints*, 2015.

[65] Y. Li, H. Yang, E. Martin, K. Ho, and L. Ying. Butterfly factorization. *ArXiv e-prints*, 2015.

[66] P.-L. Lions. On the Schwarz alternating method II. In Tony Chan, Roland Glowinski, Jacques Periaux, and Olof Widlund, editors, *Domain Decomposition Methods*, Lecture Notes in Computational Science and Engineering, pages 47–70. SIAM, 1989.

[67] F. Liu and L. Ying. Additive sweeping preconditioner for the Helmholtz equation. *ArXiv e-prints*, 2015.

[68] F. Liu and L. Ying. Recursive sweeping preconditioner for the 3D Helmholtz equation. *ArXiv e-prints*, 2015.

[69] S. Luo, J. Qian, and R. Burridge. Fast Huygens sweeping methods for Helmholtz equations in inhomogeneous media in the high frequency regime. *Journal of Computational Physics*, 270(0):378–401, 2014.

[70] F. Magoules, K. Meerbergen, and J.-P. Coyette. Application of a domain decomposition with Lagrange multipliers to acoustic problems arising from the automotive industry. *Journal of Computational Acoustics*, 08(03):503–521, 2000.

[71] G. Martin, R. Wiley, and K. Marfurt. An elastic upgrade for Marmousi. *The Leading Edge, Society for Exploration Geophysics*, 25, 2006.

[72] P.G. Martinsson and V. Rokhlin. A fast direct solver for scattering problems involving elongated structures. *Journal of Computational Physics*, 221(1):288–302, 2007.

[73] P.G. Martinsson, V. Rokhlin, and M Tygert. A randomized algorithm for the decomposition of matrices. *Applied and Computational Harmonic Analysis*, 30(1):47–68, 2011.

[74] L. C. McInnes, R. F. Susan-Resiga, D. E. Keyes, and H. M. Atassi. Additive schwarz methods with nonreflecting boundary conditions for the parallel computation of Helmholtz

problems. *Contemporary Mathematics*, 218:325–333, 1998.

[75] W. McLean. *Strongly Elliptic Systems and Boundary Integral Equations*. Cambridge University Press, 2000.

[76] A. Moiola, R. Hiptmair, and I. Perugia. Plane wave approximation of homogeneous Helmholtz solutions. *Zeitschrift für angewandte Mathematik und Physik*, 62(5):809–837, 2011.

[77] A. Moiola and E. Spence. Is the Helmholtz equation really sign-indefinite? *SIAM Review*, 56(2):274–312, 2014.

[78] P. Monk and D.-Q. Wang. A least-squares method for the Helmholtz equation. *Computer Methods in Applied Mechanics and Engineering*, 175(12):121–136, 1999.

[79] R.-E. Plessix and W. A. Mulder. Separation-of-variables as a preconditioner for an iterative Helmholtz solver. *Applied Numerical Mathematics*, 44(3):385–400, 2003.

[80] J. Poulson, B. Engquist, S. Li, and L. Ying. A parallel sweeping preconditioner for heterogeneous 3D Helmholtz equations. *SIAM Journal on Scientific Computing*, 35(3):C194–C212, 2013.

[81] J. Qian, S. Luo, and R. Burridge. Fast Huygens' sweeping methods for multiarrival Green's functions of Helmholtz equations in the high-frequency regime. *Geophysics*, 80(2):T91–T100, 2015.

[82] G. Rizzuti and W.A. Mulder. Multigrid-based shifted-laplacian preconditioning for the time-harmonic elastic wave equation. *Journal of Computational Physics*, 317:47 – 65, 2016.

[83] F.-H. Rouet, X. S. Li, P. Ghysels, and A. Napov. A distributed-memory package for dense Hierarchically Semi-Separable matrix computations using randomization. *ArXiv e-prints*, 2015.

[84] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, second edition, 2003.

[85] Y. Saad and M. H. Schultz. Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7(3):856–869, July 1986.

[86] H. A. Schwarz. Uber einen grenzubergang durch alternierendes verfahren. *Vierteljahrsschrift der Naturforschenden Gesellschaft in Zurich*, 15:272–286, 1870.

[87] A. H. Sheikh, D. Lahaye, and C. Vuik. On the convergence of shifted Laplace preconditioner combined with multilevel deflation. *Numerical Linear Algebra with Applications*, 20(4):645–662, 2013.

[88] C. Stolk. A rapidly converging domain decomposition method for the Helmholtz equation. *Journal of Computational Physics*, 241(0):240–252, 2013.

[89] C. C. Stolk. An improved sweeping domain decomposition preconditioner for the Helmholtz equation. *Advances in Computational Mathematics*, pages 1–32, 2016.

[90] A. Toselli and O. B. Windlund. *Domain Decomposition Methods   Algorithms and Theory*, volume 34 of *Springer Series in Computational Mathematics*. Springer Berlin Heidelberg, 2005.

[91] L. N. Trefethen and D. Bau. *Numerical linear algebra*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997.

[92] P. Tsuji, B. Engquist, and L. Ying. A sweeping preconditioner for time-harmonic Maxwell's equations with finite elements. *Journal of Computational Physics*, 231(9):3770 – 3783, 2012.

[93] P. Tsuji, J. Poulson, B. Engquist, and L. Ying. Sweeping preconditioners for elastic wave propagation with spectral element methods. *ESAIM: Mathematical Modelling and Numerical Analysis*, 48:433–447, 3 2014.

[94] P. Tsuji and L. Ying. A sweeping preconditioner for Yee's finite difference approximation of time-harmonic Maxwell's equations. *Frontiers of Mathematics in China*, 7(2):347–363, 2012.

[95] H. A. van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 13(2):631–644, 1992.

[96] A. Vion, R. Bélanger-Rioux, L. Demanet, and C. Geuzaine. A DDM double sweep preconditioner for the Helmholtz equation with matrix probing of the dtn map. In *Mathematical and Numerical Aspects of Wave Propagation WAVES 2013*, 2013.

[97] A. Vion and C. Geuzaine. Double sweep preconditioner for optimized Schwarz methods applied to the Helmholtz problem. *Journal of Computational Physics*, 266(0):171–190, 2014.

[98] A. Vion and C. Geuzaine. *Parallel Double Sweep Preconditioner for the Optimized Schwarz Algorithm Applied to High Frequency Helmholtz and Maxwell Equations*, pages 239–247. Springer International Publishing, Cham, 2016.

[99] S. Wang, X. S. Li, Xia J., Y. Situ, and M. V. de Hoop. Efficient scalable algorithms for solving dense linear systems with hierarchically semiseparable structures. *SIAM Journal*

on Scientific Computing, 35(6):C519–C544, 2013.

[100] X. Wang and W. W. Symes. Harmonic coordinate finite element method for acoustic waves. In *SEG Technical Program Expanded Abstracts 2012*, pages 1–5.

[101] L. Zepeda-Núñez. *Fast and scalable solvers for the Helmholtz equation*. PhD thesis, Massachusetts Institute of Technology, Cambridge MA, USA, 2015.

[102] L. Zepeda-Núñez and L. Demanet. A short note on the nested-sweep polarized traces method for the 2D Helmholtz equation. In *SEG Technical Program Expanded Abstracts 2015*, pages 3682–3687.

[103] L. Zepeda-Núñez and L. Demanet. The method of polarized traces for the 2D Helmholtz equation. *Journal of Computational Physics*, 308:347 – 388, 2016.

[104] L. Zepeda-Núñez, R. J. Hewett, and L. Demanet. Preconditioning the 2D Helmholtz equation with polarized traces. In *SEG Technical Program Expanded Abstracts 2014*, pages 3465–3470, 2014.