Fast and Scalable Solvers for the Helmholtz Equation

by

Leonardo Andrés Zepeda-Núñez

M.Sc., École Polytechnique (2010)
Diploma, École Polytechnique (2010)

Submitted to the Department of Mathematics
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2015

Author . . . .
Department of Mathematics
May 7, 2015

Certified by . . . .
Laurent Demanet
Associate Professor
Thesis Supervisor

Accepted by . . . .
Michel Goemans
Chairman, Department Committee on Graduate Theses

Fast and Scalable Solvers for the Helmholtz Equation
by
Leonardo Andrés Zepeda-Núñez

Submitted to the Department of Mathematics
on May 7, 2015, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

In this thesis we develop a new family of fast and scalable algorithms to solve the 2D high-frequency Helmholtz equation in heterogeneous medium. The algorithms rely on a layered domain decomposition and a coupling between subdomains using the Green's representation formula, which reduces the problem to a boundary integral system at the interfaces between subdomains. Simultaneously, we introduce a polarization of the waves in up- and down-going components using incomplete Green's integrals, which induces another equivalent boundary integral formulation that is easy to precondition.

The computation is divided in two stages: an offline stage, a computationally expensive but embarrassingly parallel precomputation performed only once; and an online stage, a highly parallel computation with low complexity performed for each right-hand side.

The computational efficiency of the algorithms is achieved by shifting most of the computational burden to an offline precomputation, and by reducing the sequential bottleneck in the online stage using an efficient preconditioner, based on the polarized decomposition, coupled with compressed linear algebra. The resulting algorithms have online runtime $\mathcal{O}(N/P)$, where N is the number of unknowns, and P is the number of nodes in a distributed memory environment; provided that $P = \mathcal{O}(N^\alpha)$. Typically $\alpha = 1/5$ or $1/8$.

Thesis Supervisor: Laurent Demanet
Title: Associate Professor

Acknowledgments

I would like to thank my advisor, Laurent Demanet, for his advice, support, and more importantly, patience throughout my time as a graduate student.

I would like to acknowledge my thesis committee, Alex Barnett, Maarten de Hoop, Steven Johnson and Laurent Demanet, for going through the grueling process of reading, commenting, and correcting this manuscript. I really appreciated their time, insight, and expertise.

I would like to thank my quals committee, Laurent, Steven and Michael Eichmair.

I would like to thank Oscar Bruno, for convincing me to cross the Atlantic for my Ph.D. and for introducing me to computational wave propagation. I would like to acknowledge all the professors from which I took classes at MIT, in particular Alison Malcolm, for making me discover the beauty of computational geophysics, and Richard Melrose for his excellent insight in theoretical wave propagation.

Thanks to my friends, to everyone that have already left and the ones that will remain once I have left Cambridge. I would like to thank Jose and Geannina for taking care of me, and the Chilean and French community for being a moral support during all these years. Thanks to all my friends worldwide who have opened the doors of their homes: Geoffrey, David, Damien, Meriem, Rachele, Bogdan, Clement, among many others; you are always welcome wherever I will be.

Thanks to everyone with whom I spent long hours working together with: Augustin, Rosalie, Jiawei, Ali, Martina, Russell, Vincent, Bram, David and David.

I want to thank all the staff in the Math department at MIT for making my life easier, specially Tony, Barbara, Michele, and Ali.

I would like to thank my uncle Sergio and my aunt Veronica, who introduced me to the world of mathematics.

My wholeheartedly thanks to my family and to Annika, who have always been with me, especially when I needed them the most. They always gave me a small nudge to get me back on my feet.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The main objectives of this thesis are the study and development of computationally efficient numerical methods to solve the Helmholtz equation,

$$-\triangle u(\mathbf{x}) - \frac{\omega^2}{c^2(\mathbf{x})} u(\mathbf{x}) = f(\mathbf{x}), \tag{1.1}$$

with absorbing boundary conditions. The Helmholtz equation arises from the temporal Fourier transform of the constant density acoustic wave equation, a good model for acoustic waves, and in some setting for elastic and electromagnetic waves as well. In Eq. 1.1, for a frequency ω, we seek the wavefield u generated by a general volumetric source f, in a medium in which the waves propagate at speed c.

When solving the Helmholtz equation numerically, we distinguish two radically different regimes that depend on the scaling between the frequency ω and the number of degrees of freedom of the discretized system:

- the low-frequency regime, i.e., when the frequency is kept constant as the number of degrees of freedom is increased;

- and the high-frequency regime, i.e., when the number of degrees of freedom scales[1] with the frequency.

In the low-frequency regime, Eq. 1.1 can be solved efficiently using standard numerical methods for solving elliptic equations. In the high-frequency regime, efficient methods tend to rely on extra knowledge about the structure of the wave speed.

If the wave speed c is piecewise constant with smooth boundaries, then the problem can be reduced to an integral equation posed on the interfaces. Integral formulations of the second kind (see [43]) coupled with fast summation methods provide accurate and fast algorithms (see [39, 57, 48]) to solve the high-frequency Helmholtz equation.

On the other hand, if the wave speed c is heterogeneous or piecewise constant with rough boundaries, solving the high-frequency Helmholtz equation efficiently is still an area of active research. In particular, the remaining questions concern either the

[1]The precise scaling depends on the discretization used.

accuracy, i.e., obtaining an accurate answer in a rough medium; and computational efficiency, i.e., computing the approximate solution in a fast and scalable manner.

Computing the numerical solution to the Helmholtz equation for a heterogeneous medium is of prime importance in seismic imaging. The widespread use of inversion techniques, such as full-waveform inversion (FWI), has been a major motivation for developing new highly-efficient algorithms for solving the Helmholtz equation. FWI requires the solution of the Helmholtz equation for large domains, with a possibly rough velocity, at high frequency, and for many right-hand sides. Within this context, complexity and scalability both play an important role. An algorithm needs to have a low complexity and needs to be suitable for implementation in large computing clusters. Moreover, it needs to be flexible enough to take advantage of highly optimized low-level libraries and to support new hardware architectures. In practice, there is no method currently capable of solving the Helmholtz equation for realistic industrial applications, in which the discretized systems typically have of the order of 10^10 unknowns, around 400 wavelength across the domain computed in a rough wave speed for 10^4 different right-hand sides.

In this work we primarily deal with the question of complexity and scalability in the context relevant to FWI, i.e., heterogeneous wave speed in the absence of large resonant cavities. We argue that using domain decomposition with high-quality transmission-absorbing conditions at the interfaces between subdomains, coupled with fast summation methods is the right mix of ideas to obtain a fast and highly scalable algorithm to solve the Helmholtz equation in the high-frequency regime.

We present the method of polarized traces, a fast and asymptotically scalable algorithm for the high-frequency Helmholtz equation. The method of polarized traces has an online runtime of $\mathcal{O}(N/P)$, where N is the number of degrees of freedom and P is the number of processors, provided that $P = \mathcal{O}(N^\alpha)$ for $\alpha = 1/5$. The runtimes are empirical but based on an analysis of the ranks of the off-diagonal blocks of the high-frequency Green's function.

To the author's knowledge, this work is the first instance in which such scalings have been reported. Although we deal mostly with the 2D high-frequency Helmholtz equation, the ideas within this thesis are easily extended to 3D and they provide a novel and powerful framework to develop fast and highly scalable solvers for the high-frequency Helmholtz equation within the scope of realistic industrial applications.

1.1 The Helmholtz equation

Although the Helmholtz equation models the time-harmonic propagation of acoustic waves, in some settings it can be a good approximation for time-harmonic elastic and electromagnetic waves. For the elastic wave equation, it is known that the Helmholtz decomposition enables to decouple elastic waves in p-waves and s-waves. In particular, p-waves are the curl-free gradient of a scalar potential, which satisfy an acoustic wave equation. If the medium is a fluid, only p-waves propagate, which are easily modeled in frequency using the Helmholtz equation via the scalar potential.

For solutions of Maxwell's equations, it is well known that in 2D, an electromagnetic wave can be decomposed into a transverse magnetic and a transverse electric component. With this decomposition, Maxwell's equations reduce to the Helmholtz equation for the z-component of either of the transverse components.

Although this work primarily addresses numerical analysis, let us provide a few references on known theoretical properties of the Helmholtz equation. The classical mathematical questions of existence and uniqueness of the Helmholtz equation have been settled long ago, using Fredholm or Vekua theory [101], or energy estimates via the time-dependent problem [92]. However, such results do not provide coercivity estimates with an explicit dependence on the frequency, which has resulted in a renewed interested in obtaining better estimates [16, 102, 124, 125, 126].

In the scope of numerical analysis, a great effort has concentrated on obtaining well conditioned formulations (see [15] and references therein), pollution free formulations with theoretical guarantees (see [77, 83]), and high-order discretizations (see [84] and references therein).

Finally, we mention the work of Engquist and Zhao [61], which provides upper and lower bounds for the approximate separability for the Green's functions for the Helmholtz equation in the high-frequency regime.

1.2 Applications

1.2.1 Inverse problems

The main motivating application for this thesis is inverse problems, in particular, inverse problems in exploration geophysics.

In the scope of inverse problems, we can define two problems, the forward and the inverse problem. The forward, or direct, problem consists in computing the solution to a partial differential equation when the coefficients are known. The inverse problem consists in recovering the coefficients of the PDE from partial information of the solution.

In geophysics the forward problem is defined as follows: given the physical properties of the medium, in this case the wave speed $c(\mathbf{x})$ as function of $\mathbf{x} = (x, y, z)$, and the location of the source, compute the wavefield u at the surface. On the other hand, the inverse problem is to compute the wave speed from the location of the sources and the wavefields measured at the surface. One standard technique for solving the inverse problem is full-waveform inversion (FWI), which recasts the inverse problem as a minimization problem. The objective functional to minimize is the least-squares misfit between the real and simulated data.

Even though the concept of FWI has been around since the mid 1980's with the seminal work of Tarantola in 1984 [134], it was not until Pratt's work in 1999 [113], that the method became popular. In particular, Pratt used a multi-scale approach to avoid the local minima during the optimization routine. This multi-scale approach is based on frequency sweeps: using Parseval's Theorem the minimization is posed in the frequency domain, processing the data from low to high frequencies (see [35, 120]). Moreover, this approach is efficient given that only a handful of frequencies are needed for inversion.

Although FWI has become increasingly popular, it is not as widespread in industry as would naively be expected. This is mainly due to its ill-posedness, the non-convexity of the minimization objective functional and its prohibitively large computational cost. Great effort has been devoted to regularize the problem, to alleviate the ill-possessedness, and to convexify the objective functional (see, for example, [133]). In addition, the reduction of the computational cost, to which this thesis is closely related, has been received much attention lately (see Section 1.4).

In fact, waves have a particularly favorable feature for inverse problems: they propagate singularities (see [82, 135]). This allows the information to travel long distances, which is extremely helpful when only boundary data is available, as in the case of seismic imaging. Alas, the propagation of singularities makes the forward problem, in which FWI relies, extremely difficult to solve efficiently. This difficulty owes to the propagation of singularities that imposes that any solver needs to be sequential, which appears to be in contradiction with modern computer architectures, which tend to be highly parallel.

Although this thesis has focused on applications to geophysics, an efficient Helmholtz solver is of great interest in any wave-based inverse problem, given that most of them rely on the ability to solve the forward problem efficiently. In this category we have biomedical applications, underwater acoustics, non-destructive material testing, non-destructive testing for cracks, ground penetrating radar, sonar sensing, among many others.

1.2.2 Optimal design

Also within the scope of geophysics, one of the most compelling applications of fast Helmholtz solvers is optimal survey design. In marine seismic imaging, the first step of the inversion is the acquisition of the data through a survey. The data acquisition needs expensive and highly-specialized equipment; furthermore, it is an expensive and potentially dangerous operation. In a nutshell, a survey consist of one of more vessels equipped with air cannons to produce seismic waves, towing large arrays of receivers, which record the echoes of the seismic waves being reflected from the different formations in the subsurface.

In general, one may have *a priori* knowledge of the area. This knowledge can be a geological map, or an estimation of the wave speed from another study. Based on this information one may want to focus the exploration in one particular region of the subsurface. In such situations, we want to obtain an optimal survey, i.e., the number and kind of vessels involved and the path they need follow to minimize the time spent in the survey and to maximize the illumination of the region of interest. Illumination, in this context, can be broadly understood as the the amount of acoustic energy reflected from the particular area of interest that can be captured by the receivers being towed. Within the optimization routine, one needs to solve the Helmholtz equation several times with sources located on the possible paths of the vessels to obtain the desired illumination.

Once again, even though the objective application arises from industry needs in exploration geophysics, the range of optimal design problems related to waves is vast. For civilian applications we can list, for example, the design of cellphone antennas, the design of more powerful radars, the design of quieter airliners engines, optimal focusing of ultrasound beam for non-invasive intra-cranial surgery and tumor ablation, and the modeling and design of resonators in nanophotonics, among many others. In military applications, we can list reduction of the radar cross-section, design of electromagnetic shielding, etc.

1.3 Results

1.3.1 Sublinear Helmholtz solver

We present in Chapter 2 the method of polarized traces, a fast and asymptotically scalable solver for the high-frequency Helmholtz equation with an online runtime $\mathcal{O}(N/P)$, provided[2] that $P = \mathcal{O}(N^{1/8})$. The method combines:

- finite-differences discretization of the differential operator in the volume,
- domain decomposition in layers,
- exact transmission conditions based on the discrete Green's representation formula, and
- high-quality absorbing boundary conditions, in the form of perfectly matched layers (PML), between the interfaces of the subdomains;

to reduce the problem in the volume to a discrete integral boundary problem at the interfaces, based on the discrete Green's functions arising from the finite differences-discretization.

Using the transmission and absorbing conditions we define approximate polarizing conditions that provide a natural decomposition of the wavefield into up- and down-going components. The polarizing conditions, coupled with a detailed analysis of the discrete integral boundary equation, are used to design a highly efficient preconditioner for the integral boundary equation. The preconditioner is based on an application of Green's integrals to interface data, which can be highly compressed using standard $\mathcal{H}$-matrices (see [20]), yielding a fast application and a fast solve.

The method has two stages: an offline stage, which is computationally expensive but embarrassingly parallel and performed only once; and an online stage, which is performed for each right-hand side, but has low complexity, which results in fast online runtimes. When the number of different right-hand sides is large, the fast online solves offset the expensive offline stage. The fast online runtimes are achieved by shifting most of the computational burden to a parallel computation, and reducing most of the sequential bottleneck using the efficient preconditioner and compressed matrix-algebra mentioned earlier.

[2]We show first how to obtain an algorithm with online runtime $\mathcal{O}(N/P)$ provided that $P = \mathcal{O}(N^{1/8})$, then we present a variant with the less stringent condition of $P = \mathcal{O}(N^{1/5})$.

1.3.2 Extensions

The method of polarized traces introduced in Chapter 2 is an efficient iterative solver for the Helmholtz equation; however, it has two main drawbacks. The complexity deteriorates greatly in the case of large resonant cavities and sharp contrasts; and the offline computational burden can be prohibitively expensive for large problems.

In Chapter 3, we alleviate these issues by introducing two different variants of the method of polarized traces.

- First, we present a variant of the boundary integral equation presented in Chapter 2, that we solve using a compressed-block LU approach. We obtain a direct compressed method that has the same online complexity as the method of polarized traces, even for media featuring large resonant cavities, with a more thorough offline precomputation.

  This algorithm is particular well suited for situations in which a large amount of fast solves for the same matrix is needed, such as optimal focusing for non-invasive intra-cranial surgery, optimal survey design, optimal focusing for ultrasound tumor ablation, among others.

- Second, we present a nested sweeps extension to the method of polarized traces, with improved asymptotic runtimes in a distributed memory environment. The method has an online complexity of $\mathcal{O}(N/P)$ provided that $P = \mathcal{O}(N^{1/5})$, which is an improvement with respect to the more stringent condition of $P = \mathcal{O}(N^{1/8})$ for the method of polarized traces. The complexity holds even in the presence of rough media of geophysical interest. Moreover, its performance is completely agnostic to the source.

  The gains in complexity are mainly due to the nested layered partitioning and a special factorization of the Green's integrals that allows for a more efficient parallelization.

  This algorithm would especially be of interest in the context of time-lapse full-waveform inversion, continuum reservoir monitoring, and local inversion. Thanks to the nested layered partition, if the update to the model is localized, then most the precomputation can be re-used. The only extra cost is the refactorization and computation of the Green's functions in the subdomains with a non null intersection with the support of the update, greatly reducing the computational cost.

  We point out that this approach can be further parallelized using distributed algebra libraries. Moreover, the sweeps can be pipelined to maintain a constant load among all the nodes.

1.3.3 Harmonic Extrapolation

The numerical methods presented in this thesis are based on the same layered (or nested layered) partitioning of the domain. The algorithms hinge on the fact that if the wavefield and its normal derivative are known at one interface between layers, then it is possible to compute the wavefield at the neighboring interfaces in a stable and fast manner. From an algebraic point of view this corresponds to an ordered elimination of unknowns running a three term recurrence. Analytically, this operation can be understood as a depth extrapolation[3] with full aperture data, i.e. data in the whole real axis.

In Chapter 4 we study the analytical counterpart of the elimination of unknowns. In particular, we study the extrapolation properties of waves in a homogeneous medium with partial data, i.e., data supported on a segment instead of the full real axis.

We show that, provided that the wavefield is one-way, and its restriction is bandlimited, then it is possible to extrapolate the wavefield, with partial data supported on a segment, in a stable manner in the direction orthogonal to the segment. The extrapolation is accurate inside a cone supported on the segment, whose aperture is linked to the bandlimit of the trace data.

1.4 Related work

As mentioned earlier, domain decomposition with accurate transmission and absorbing boundary conditions is the right mix of ideas to solving the high-frequency Helmholtz equation. We provide a succinct overview of the recent work on domain decomposition methods for the Helmholtz equation:

- The earliest suggestion to perform domain decomposition with transmission boundary conditions is perhaps to be found in the work of Després [54], which led, in joint work with Cessenat and Benamou, to the ultra weak variational formulation [11, 29, 30, 31];

- Their work spawned a series of new formulations within the framework of Discontinuous Galerkin methods, such as the Trefftz formulation of Perugia et al. [100], the plane wave discontinuous Galerkin method [76, 80] and the discontinuous enrichment method of Farhat et al. [66];

- Simultaneously to the ultra weak formulation, the partition of unity method (PUM) by Babuska and Melenk [5], and the least-squares method by Monk and Wang [103], were developed.

- A concise and good review of domain decomposition method with non-polynomial basis can be found in [7], which includes a high-quality MATLAB toolbox for computing scattering from polygons;

- Gander and Nataf introduced the AILU method in 2001 [69, 71];

[3]For a more complete exposition of wave extrapolation methods see [51, 118, 129] and references therein.

- Meanwhile, finite elements tear and interconnect (FETI) methods were developed; a specialized version of the FETI method for the Helmholtz equation was developed by Farhat et al. in [67, 65];

- Hiptmair et al. proposed the multi-trace formulation [79], which involves solving an integral equation posed on the interfaces of a decomposed domain, which is naturally well suited for operator preconditionning in the case of piece-wise constant medium;

- Plessix and Mulder proposed a method similar to AILU in 2003 [109];

- One of the earlier methods using absorbing boundary conditions between subdomains can be found in the work of Engquist and Zhao [60];

- The first linear complexity claim was made in the work of Engquist and Ying on sweeping preconditioners in 2011 [58, 59], using a special kind of domain decomposition into grid-spacing-thin layers;

- Stolk in 2013 [128] presented a domain decomposition method, with a complexity comparable to the sweeping preconditioners, that relies on single layer potentials to transfer the information between subdomains;

- More recently, Geuzaine and Vion explored approximate transmission boundary conditions [143, 142] coupled with a multiplicative Schwartz iteration, to improve on traditional domain decomposition methods;

- Chen and Xiang proposed another instance of efficient domain decomposition where the emphasis is on transferring sources from one subdomain to another [37, 38];

- Luo et al. proposed a large-subdomain sweeping method, based on an approximate Green's function by geometric optics and the butterfly algorithm, which can handle transmitted waves in very favorable complexity [94]; and a variant of this approach that can handle caustics in the geometric optics Ansatz [114];

- Conen et al. developed an additive Schwarz iteration coupled with coarse grid preconditioner based on an eigenvalue decomposition of the DtN maps inside each subdomain [44];

- Poulson et al. parallelized the sweeping preconditioners in 3D to deal with very large-scale problems in geophysics [112]. Tsuji et al. designed a spectrally accurate sweeping preconditioner for time-harmonic elastic waves [139], and time-harmonic Maxwell equations [138, 140].

- Liu and Ying recently developed a recursive version of the sweeping preconditioner in 3D that decreases the off-line cost to linear complexity [93];

- Barnett et al. [8] developed an accurate 2D Helmholtz solver for stratified media using integral equations via an accurate evaluation of the Green's function.

Finally, Gander and Zhang performed a benchmark for small 3D problem with a challenging geophysical model using different domain decomposition methods based on local transmission conditions in [72].

Most of the methods mentioned above use either physical, or analytical information to precondition the system, making them highly tailored for the Helmholtz equation. On the other hand, purely algebraic methods, such a the incomplete LU preconditioner, are more general, albeit providing suboptimal results for the Helmholtz problem. In this direction, some extensions to the incomplete LU methods have been proposed lately, complementing the work of Saad et al. [41] and Benzi et al. [12]. The algorithms in this category hinge on using a fast direct method as a preconditioner. A good example, for elliptic problems, is the recent work of Aminfar, Ambikasaran and Darve [2, 3] that uses low tolerance $\mathcal{H}$-matrices to compute an approximate inverse, which is used as a preconditioner. Along the same lines, but using sparse matrices, we find the work of Saad et al. [104] and the work of Byckling and Huhtanen [26]. We point out that, although these techniques work well for elliptic problem with sparse matrices, they are suboptimal when applied to the high-frequency Helmholtz equation.

In a different 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 [50, 144], which couples multi-frontal elimination with $\mathcal{H}$-matrices (see [74, 55] for the nested dissection reordering and multi-frontal methods). Another example is the work of Gillman, Barnett and Martinsson on computing impedance-to-impedance maps in a multiscale fashion [75], and the work of Bremer, Gillmann and Martinsson on recursive construction of scattering matrices [24]. In the scope of integral equations, Lai, Ambikasaran and Greengard proposed a direct method based on $\mathcal{H}$-matrices to solve a scattering problem from large cavities in [88]. Finally, Cho and Barnett [40] proposed a robust fast direct method based on integral equations to solve the scattering from a quasi-periodic layered medium. It is not yet clear whether offline linear complexity scalings can be achieved this way, though good direct methods are often faster in practice than the iterative methods mentioned above. The main issue with direct methods is the lack of scalability to very large-scale problems due to the memory requirements.

Some progress has also been achieved with multigrid methods, though the complexity scalings do not appear to be optimal in two and higher dimensions:

- Bhowmik and Stolk recently proposed new rules of optimized coarse grid corrections for a two level multigrid method [131, 130];
- Brandt and Livshits developed the wave-ray method [23], in which the oscillatory error components are eliminated by a ray-cycle, that exploits a geometric optics approximation of the Green's function;
- Elman and Ernst use a relaxed multigrid method as a preconditioner for an outer Krylov iteration in [56];
- Haber and MacLachlan proposed an alternative formulation for the Helmholtz equation [78], which reduces the problem to solve an eikonal equation and

an advection-diffusion-reaction equation, which can be solved efficiently by a Krylov method using a multigrid method as a preconditioner; and

- Grote and Schenk [21] proposed an algebraic multi-level preconditioner for the Helmholtz equation.

Erlangga et al. [63] showed how to implement a simple, although suboptimal, complex-shifted Laplace preconditioner with multigrid, which shines for its simplicity, and whose performance depends on the choice of the complex-shift. A suboptimal multigrid method with multilevel complex shifts was implemented in 2D and 3D by Cools, Reps and Vanroose [45]. Another variant of complex-shifted Laplacian method with deflation was studied by Sheikh et al. [119]. The choice of the optimal complex-shift has been studied by Cools and Vanroose [46] and by Gander et al. [73]. Finally, some extensions of the complex-shifted Laplacian preconditioner have been proposed recently by Cools and Vanroose [47].

Chen et al. implemented a 3D solver using the complex-shifted Laplacian with optimal grids to minimize pollution effects in [36]. Multigrid methods applied to large 3D examples have been implemented by Plessix [108], Riyanti et al. [115] and, more recently, Calandra et al. [27]. A good review of iterative methods for the Helmholtz equation, including complex-shifted Laplace preconditioners, is in [62]. Another review paper that discussed the difficulties generated by the high-frequency limit is [64]. Finally, beautiful mathematical expositions of the Helmholtz equation and a good reviews are [102] and [33].

Chapter 2

The method of polarized traces

Many recent papers have shown that domain decomposition with accurate transmission boundary conditions is the right mix of ideas for simulating propagating high-frequency waves in a heterogeneous medium. To a great extent, the approach can be traced back to the AILU method of Gander and Nataf in 2001 [69, 71]. The first linear complexity claim was perhaps made in the work of Engquist and Ying on sweeping preconditioners in 2011 [58, 59] – a special kind of domain decomposition into grid-spacing-thin layers. In 2013, Stolk [128] restored the flexibilty to consider coarser layerings, with a domain decomposition method that realizes interface transmission via an ingenious forcing term, homologous to a single layer potential, resulting in linear complexity scalings very similar to those of the sweeping preconditioners. Other authors have since then proposed related methods, including [37, 143], which we reviewed in section 1.4. Many of these references present isolated instances of what should eventually become a systematic understanding of how to couple absorption/transmission conditions with domain decomposition.

In a different 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 [50], which couples multi-frontal elimination with $\mathcal{H}$-matrices. Another example is the work of Gillman, Barnett and Martinsson on computing impedance-to-impedance maps in a multiscale fashion [75]. It is not yet clear whether offline linear complexity scalings can be achieved this way, though good direct methods are often faster in practice than the iterative methods mentioned above. The main issue with direct methods is the lack of scalability to very large-scale problems due to the memory requirements and a costly communication overhead in distributed memory environments.

The method for solving the 2D high-frequency Helmholtz equation presented in this chapter is a hybrid: it uses legacy direct solvers locally on large subdomains, and shows how to properly couple those subdomains with accurate transmission in the form of an incomplete Green's representation formula acting on polarized (i.e., one-way) waves. The novelty of this method is twofold:

- We show how to reduce the discrete Helmholtz equation to an integral system at interfaces in a way that does not involve Schur complements, but which allows for polarization into one-way components;

- We show that when using fast algorithms for the application of the integral kernels, the online time complexity of the numerical method can become *sublinear in* N in a parallel environment, i.e., $\mathcal{O}(N/L)$ over $L \ll N$ nodes[1].

The proposed numerical method uses a layering of the computational domain, and also lets L be the number of layers. It morally reduces to a sweeping preconditioner when there are as many layers as grid points in one direction ($L = n \sim N^{1/2}$); and it reduces to an efficient direct method when there is no layering ($L = 1$). In both those limits, the online complexity reaches $\mathcal{O}(N)$ up to log factors.

But it is only when the number of layers L obeys $1 \ll L \ll N$ that the method's online asymptotic complexity scaling $\mathcal{O}(N/L)$ is strictly better than $\mathcal{O}(N)$. In retrospect, this example shows that the goal of reaching linear sequential $\mathcal{O}(N)$ complexity (e.g. through a number of iterations independent of the frequency) was not ambitious enough for the 2D Helmholtz equation.

2.1 Rationale and results

We provide the rationale behind the method of polarized traces using standard arguments from integral equations and we introduce the concept of polarization. Then, we give an intuitive glimpse of how the algorithm works and provide the main complexity claims for the method.

2.1.1 Polarization

In the context of scalar waves in an unbounded domain, we say that a wave is *polarized* at an interface when it is generated by sources supported only on one side of that interface.

Polarization is key to localizing propagating waves. If, for instance, the Helmholtz equation is posed with absorbing conditions in a large domain Ω, but with sources in some small subdomain $\Omega_1 \subset \Omega$, then the exterior unknowns can be eliminated to yield a local problem in Ω_1 by an adequate choice of "polarizing" boundary condition. Such a boundary condition can take the form $\frac{\partial u}{\partial n} = Du$, where D is the Dirichlet-to-Neumann map exterior to Ω_1. In a finite difference framework, D can be represented via a Schur complement of the unknowns in the exterior region $\Omega \setminus \Omega_1$. See figure 2-1 for a sample geometry where $\Omega_1 = \Omega^{\text{down}}$, the bottom half-plane.

A polarizing boundary condition at $\partial\Omega_1$ may be absorbing in some special cases (such as a uniform medium), but is otherwise more accurately described as the *image* of the absorbing boundary condition at $\partial\Omega$ by reduction as outlined above. Polarized waves at $\partial\Omega_1$ – as we defined them – are not in general strictly speaking outgoing, because of the presence of possible scatterers in the exterior zone $\Omega \setminus \Omega_1$. These scatterers will generate reflected/incoming waves that need to be accounted for in the polarization condition at $\partial\Omega_1$.

[1]The upper bound on L is typically of the form $\mathcal{O}(N^{1/8})$, with some caveats. This discussion is covered in sections 2.1.3 and 2.6.

We say that a polarization condition is exact when the reduction of unknowns is done in the entire exterior domain, without approximation. In a domain decomposition framework, the use of such exact polarizing conditions would enable solving the Helmholtz equation in two sweeps of the domain – typically a top-down sweep followed by a bottom-up sweep [70].

However, constructing exact polarizing conditions is a difficult task. No matter the ordering, elimination of the exterior unknowns by direct linear algebra is costly and difficult to parallelize. Probing from random vectors has been proposed to alleviate this problem [10], but requires significant user intervention. Note that elimination of the *interior* unknowns by Schur complements, or equivalently computation of an interior Dirichlet-to-Neumann map, may be interesting [75], but does not result in a polarized boundary condition. Instead, this thesis explores a local formulation of approximate polarized conditions in integral form, as incomplete Green's identities involving local Green's functions.

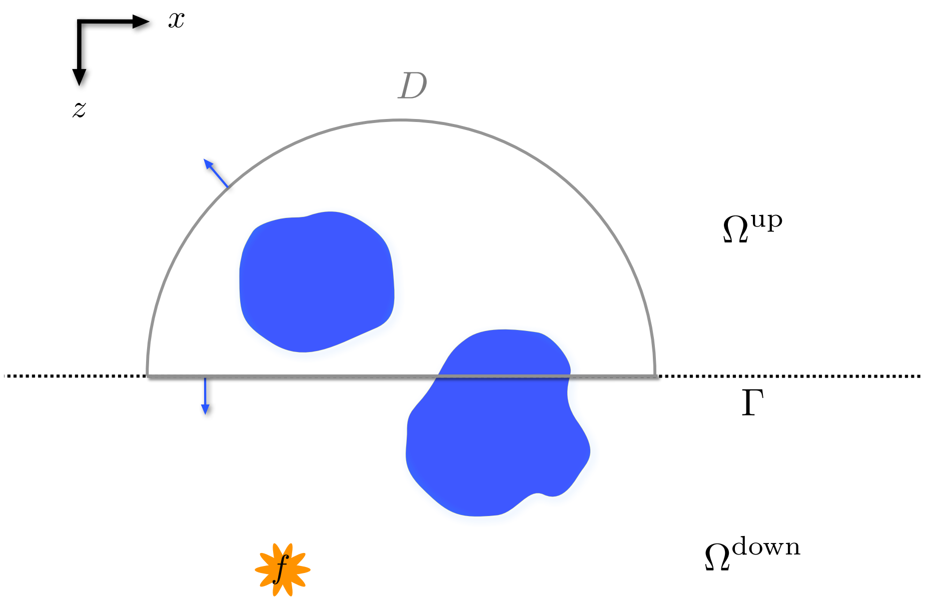


Figure 2-1: Illustration of Eqs. 2.1 and 2.2.

To see how to express a polarizing condition in boundary integral form, consider Fig. 2-1, in which we solve the 2D Helmholtz equations in a heterogeneous medium (homogeneous background with a compactly supported perturbation) with Sommerfeld radiation conditions at infinity. Let $\mathbf{x} = (x, z)$ with z pointing down. Consider a linear interface Γ partitioning $\mathbb{R}^2$ as $\Omega^{\text{down}} \cup \Omega^{\text{up}}$ (lower- and upper-half planes respectively), with $f \neq 0$ in the interior of Ω^{down}. Let $\mathbf{x} \in \Omega^{\text{up}}$, and consider a contour made up of Γ and a semi-circle D at infinity in the upper half-plane Ω^{up}. Given that 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) [87, 98] resulting in the *incomplete Green's formula*

$$u(\mathbf{x}) = \int_\Gamma \left(\frac{\partial G}{\partial z_y}(\mathbf{x},\mathbf{y})u(\mathbf{y}) - G(\mathbf{x},\mathbf{y})\frac{\partial u}{\partial z_y}(\mathbf{y}) \right) dS_{\mathbf{y}}, \qquad \mathbf{x} \in \Omega_{\text{up}} \setminus \Gamma. \tag{2.1}$$

On the contrary, if $\mathbf{x}$ approaches Γ from below, then we obtain the *annihilation formula*

$$0 = \int_\Gamma \left(\frac{\partial G}{\partial z_y}(\mathbf{x},\mathbf{y})u(\mathbf{y}) - G(\mathbf{x},\mathbf{y})\frac{\partial u}{\partial z_y}(\mathbf{y}) \right) dS_{\mathbf{y}}, \qquad \mathbf{x} \to \Gamma \text{ from below.} \tag{2.2}$$

Eqs. 2.1 and 2.2 are equivalent; either one can be used as the definition of a polarizing boundary condition on Γ. This boundary condition is exactly polarizing if G is taken as the global Green's function for the Helmholtz equation in $\mathbb{R}^2$ with SRC.[2]

Instead, this thesis proposes to use these conditions with a *local* Green's function G that arises from a problem posed in a slab around Γ, with absorbing boundary conditions at the edges of the slab. With a local G, the conditions (Eq. 2.1 or 2.2) are only approximately polarizing.

Other approximations of absorbing conditions, such as square-root operators, can be good in certain applications [51, 118, 127, 68, 129], but are often too coarse to be useful in the context of a fast domain decomposition solver. In particular, it should be noted that square-root operators do *not* approximate polarizing conditions to arbitrary accuracy in media with heterogeneity normal to the boundary.

Finally, we point out the this approach can be found for the homogeneous case under the name of Rayleigh integrals in Chapter 5 of [14], in which the polarizing condition is called a causality condition; moreover, this approach is equivalent to the upwards-propagating radiation condition (UPRC) in [32].

2.1.2 Algorithm

Let Ω be a rectangle in $\mathbb{R}^2$, and consider a layered partition of Ω into slabs, or layers $\{\Omega^\ell\}_{\ell=1}^L$. The squared slowness $m(\mathbf{x}) = 1/c(\mathbf{x})^2$, $\mathbf{x} = (x,z)$[3], is the only physical parameter we consider in this Chapter. Define the global Helmholtz operator at frequency ω as

$$\mathcal{H}u = \left(-\triangle - m\omega^2\right) u \qquad \text{in } \Omega, \tag{2.3}$$

with an absorbing boundary condition on $\partial\Omega$, realized to good approximation by a perfectly matched layer surrounding Ω. Let us define f^ℓ as the restriction of f to Ω^ℓ, i.e., $f^\ell = f\chi_{\Omega^\ell}$. Define the local Helmholtz operators as

$$\mathcal{H}^\ell u = \left(-\triangle - m\omega^2\right) u \qquad \text{in } \Omega^\ell, \tag{2.4}$$

with an absorbing boundary condition on $\partial\Omega^\ell$. Let u be the solution to $\mathcal{H}u = f$. Using the local GRF, the solution can be written without approximation in each

[2]This fact remains true in higher dimension as long as Γ is a hyper-plane of co-dimension 1.

[3]We assume that z points downwards, and that Ω^ℓ is above Ω^j, provided that $\ell < j$.

layer as

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

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

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 Eq. 2.5 as

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

The knowledge of u and $\partial_z u$ on the interfaces $\{\Gamma_{\ell,\ell+1}\}_{\ell=1}^{N-1}$ therefore suffices to recover the solution everywhere in Ω.

We further 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 Eq. 2.2), and $(u^\downarrow, \partial_z u^\downarrow)$ polarized down 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 computational unknowns for the numerical method.

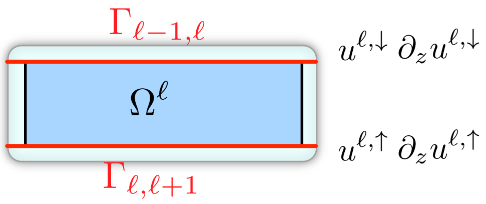

$\Gamma_{\ell-1,\ell}$
$u^{\ell,\downarrow}$ $\partial_z u^{\ell,\downarrow}$
Ω^ℓ
$u^{\ell,\uparrow}$ $\partial_z u^{\ell,\uparrow}$
$\Gamma_{\ell,\ell+1}$

Figure 2-2: Illustration of Eq. 2.6. The light-shaded layer around Ω^ℓ represents the absorbing layer.

The discrete system is then set up from algebraic reformulations of the local GRF (Eq. 2.6) with the polarizing conditions (Eqs. 2.1 and 2.2), in a manner that will be made explicit below. The reason for considering this sytem is twofold:

1. It has a 2-by-2 structure with block-triangular submatrices on the diagonal, and comparably small off-diagonal submatrices as shown in Fig. 2-3. 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 Eq. 2.1.

2. Each of these submatrices decomposes into blocks that are somewhat compressible in adaptive low-rank-partitioned format. This property stems from

the polarization conditions, and would not hold for the Schur complements of the interior of the slab.

Point 1 ensures that the number of GMRES iterations stays small and essentially bounded as a function of frequency and number of slabs. Point 2 enables the sublinear-time computation of each iteration.

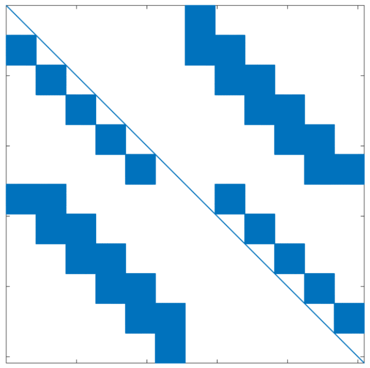


Figure 2-3: Sparsity pattern of the polarized system.

2.1.3 Complexity scalings

Time complexities that grow more slowly than the total number N of volume unknowns are possible in a L-node cluster, in the following sense: the input consists of a right-hand-side f distributed to L nodes corresponding to different layers, while the problem is considered solved when the solution is locally known for each of the L nodes/layers.

The Helmholtz equation is discretized as a linear system of the form $\mathbf{Hu} = \mathbf{f}$. There is an important distinction between

- the offline stage, which consists of any precomputation involving $\mathbf{H}$, but not $\mathbf{f}$; and
- the online stage, which involves solving $\mathbf{Hu} = \mathbf{f}$ for possibly many right-hand sides $\mathbf{f}$.

By online time complexity, we mean the runtime for solving the system once in the online stage. The distinction is important in situations like geophysical wave propagation, where offline precomputations are often amortized over the large number of system solves with the same matrix $\mathbf{H}$.

We let $n = N^{1/2}$ for the number of points per dimension (up to a constant). Since most of our complexity claims are empirical, our O notation may include some log factors. All the numerical experiments in this Chapter assume the simplest second-order finite difference scheme for constant-density, heterogeneous-wave-speed, fixed-frequency acoustic waves in 2D – a choice that we make for simplicity of the exposition.

High frequency either means $\omega \sim n$ (constant number of points per wavelength), or $\omega \sim n^{1/2}$ (the scaling for which second-order FD are expected to be accurate), with different numerical experiments covering both cases.

The table below gathers the observed complexity scalings. We assume that the time it takes to transmit a message of length m on the network is of the form $\alpha + \beta m$, with α the latency and β the inverse bandwidth.

Step	Sequential	Zero-comm parallel	Number of processors	Comm cost
offline	-	$\mathcal{O}((N/L)^{3/2})$	$N^{1/2}L$	$\mathcal{O}(\alpha L + \beta N L)$
online	$\mathcal{O}(N^{\gamma}L)$	$\mathcal{O}\left(N \log(N)/L\right)$	L	$\mathcal{O}(\alpha L + \beta N^{1/2} L)$

Table 2.1: Time complexities for the different stages of the solver. The parameter γ is striclty less than one; its value depends on the scaling of ω vs. N. A representative value is $\gamma = 3/4$ (more in the text).

These totals are decomposed over the different tasks according to the following two tables.

Step	Sequential	Zero-comm parallel	Number of processors
Factorization	-	$\mathcal{O}((N/L)^{3/2})$	L
Computation of the Green's functions	-	$\mathcal{O}\left(N \log(N)/L\right)$	L
Compression of the Green's functions	-	$\mathcal{O}(R_{\max} N \log N)$	$N^{1/2}L$

Table 2.2: Complexity of the different steps within the offline computation.

Step	Sequential	Zero-comm parallel	Number of processors
Backsubstitutions for the local solves	-	$\mathcal{O}\left(N \log(N)/L\right)$	L
Solve for the polarized traces	$\mathcal{O}(N^{\gamma}L)$	-	1
Reconstruction in the volume	-	$\mathcal{O}\left(N \log(N)/L\right)$	L

Table 2.3: Complexity of the different steps within the online computation.

The "sweet spot" for L is when $N^{\gamma}L \sim \frac{N}{L}$, i.e., when $L \sim N^{\frac{1-\gamma}{2}}$. The value of γ depends on how the frequency ω scales with n:

- When $\omega \sim n^{1/2}$ ($\mathcal{O}(n^{1/2})$ points per wavelength), the second-order finite difference scheme is expected to be accurate in a smooth medium. In that case, we observe $\gamma = \frac{5}{8}$ in our experiments, even when the medium is not smooth. Some theoretical arguments indicate, however, that this scaling may not continue to hold as $N \to \infty$, and would become $\gamma = \frac{3}{4}$ for values of N that we were not able to access computationally. Hence we prefer to claim $\gamma = \frac{3}{4}$.

- When $\omega \sim n$ ($\mathcal{O}(1)$ points per wavelength), the second-order finite difference scheme is not pointwise accurate. Still, it is a preferred regime in exploration geophysics. In that case, we observe $\gamma = \frac{7}{8}$. This relatively high value of γ is due to the poor quality of the discretization. For theoretical reasons, we anticipate that the scaling would again become $\gamma = \frac{3}{4}$ as $\omega \sim n$, were the quality of the discretization not an issue.

As the discussion above indicates, some of the scalings have heuristic justifications, but we have no mathematical proof of their validity in the case of heterogeneous media of limited smoothness. Counter-examples may exist. For instance, if the contrast increases between the smallest and the largest values taken on by m – the case of cavities and resonances – it is clear that the constants in the $\mathcal{O}$ notation degrade. See section 2.6 for a discussion of the heuristics mentioned above.

2.2 Discrete Formulation

In this section we show how to reformulate the discretized 2D Helmholtz equation, in an exact fashion, as a system for interface unknowns stemming from local Green's representation formulas.

2.2.1 Discretization

Let $\Omega = (0, L_x) \times (0, L_z)$ be a rectangular domain. Throughout this Chapter, we pose Eq. 2.3 with absorbing boundary conditions on $\partial\Omega$, realized as a perfectly matched layer (PML) [13, 85]. This approach consists of extending the computational domain with an absorbing layer, in which the differential operator is modified to efficiently damp the outgoing waves and reduce the reflections due to the truncation of the domain.

Let $\Omega^{\text{ext}} = (-\delta_{\text{pml}}, L_x + \delta_{\text{pml}}) \times (-\delta_{\text{pml}}, L_z + \delta_{\text{pml}})$ be the extended rectangular domain containing Ω and its absorbing layer. The Helmholtz operator in Eq. 2.3 then takes the form

$$\mathcal{H} = -\partial_{xx} - \partial_{zz} - m\omega^2, \qquad \text{in } \Omega^{\text{ext}}, \tag{2.7}$$

in which the differential operators are redefined following

$$\partial_x \to \frac{1}{1 + i\frac{\sigma_x(\mathbf{x})}{\omega}}\partial_x, \qquad \partial_z \to \frac{1}{1 + i\frac{\sigma_z(\mathbf{x})}{\omega}}\partial_z, \tag{2.8}$$

and where m is an extension[4] of the squared slowness. Moreover, $\sigma_x(\mathbf{x})$ is defined as

$$\sigma_x(\mathbf{x}) = \begin{cases} \frac{C}{\delta_{\text{pml}}}\left(\frac{x}{\delta_{\text{pml}}}\right)^2, & \text{if } x \in (-\delta_{\text{pml}}, 0), \\ 0, & \text{if } x \in [0, L_x], \\ \frac{C}{\delta_{\text{pml}}}\left(\frac{x-L_x}{\delta_{\text{pml}}}\right)^2, & \text{if } x \in (L_x, L_x+\delta_{\text{pml}}), \end{cases} \tag{2.9}$$

and similarly for $\sigma_z(\mathbf{x})$. We remark that δ_{pml} and C can be seen as tuning parameters. 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 media; and C is chosen to provide enough absorption.

With this notation we rewrite Eq. 2.3 as

$$\mathcal{H}u = f, \qquad \text{in } \Omega^{\text{ext}}, \tag{2.10}$$

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 Eq. 2.10. For the interior points $\mathbf{x}_{i,j} \in \Omega$, we have

$$(\mathbf{Hu})_{p,q} = \frac{1}{h^2}\left(-\mathbf{u}_{p-1,q} + 2\mathbf{u}_{p,q} - \mathbf{u}_{p+1,q}\right) + \frac{1}{h^2}\left(-\mathbf{u}_{p,q-1} + 2\mathbf{u}_{p,q} - \mathbf{u}_{p,q+1}\right) - \omega^2 m(\mathbf{x}_{p,q}). \tag{2.11}$$

In the PML, we discretize

$$\alpha_x \partial_x(\alpha_x \partial_x u) \qquad \text{as} \qquad \alpha_x(\mathbf{x}_{p,q}) \frac{\alpha(\mathbf{x}_{p+1/2,q})(\mathbf{u}_{p+1,q} - \mathbf{u}_{p,q}) - \alpha_x(\mathbf{x}_{p-1/2,q})(\mathbf{u}_{p,q} - \mathbf{u}_{p-1,q})}{h^2}, \tag{2.12}$$

$$\alpha_z \partial_z(\alpha_z \partial_z u) \qquad \text{as} \qquad \alpha_z(\mathbf{x}_{p,q}) \frac{\alpha(\mathbf{x}_{p,q+1/2})(\mathbf{u}_{p,q+1} - \mathbf{u}_{p,q}) - \alpha_z(\mathbf{x}_{p,q-1/2})(\mathbf{u}_{p,q} - \mathbf{u}_{p,q-1})}{h^2}, \tag{2.13}$$

where

$$\alpha_x(\mathbf{x}) = \frac{1}{1 + i\frac{\sigma_x(\mathbf{x})}{\omega}}, \qquad \alpha_z(\mathbf{x}) = \frac{1}{1 + i\frac{\sigma_z(\mathbf{x})}{\omega}}. \tag{2.14}$$

Finally, we solve

$$(\mathbf{Hu})_{p,q} = \mathbf{f}_{p,q}, \qquad (p,q) \in [\![-n_{\text{pml}}+1, n_x + n_{\text{pml}}]\!] \times [\![-n_{\text{pml}}+1, n_z + n_{\text{pml}}]\!], \tag{2.15}$$

for the global solution $\mathbf{u}$. We note that the matrix $\mathbf{H}$ in Eq. 2.15 is not symmetric,

[4]We assume that $m(x)$ is given to us in Ω^{ext}. If it isn't, normal extension of the values on $\partial\Omega$ is a reasonable alternative.

though there exists an equivalent symmetric (non-Hermitian) formulation of the PML [58, 59]. While the symmetric formulation presents theoretical advantages[5], working with the non-symmetric form gives rise to less cumbersome discrete Green's formulas in the sequel.

2.2.2 Domain Decomposition

We partition Ω^{ext} in a layered fashion into L subdomains $\{\Omega^\ell\}_{\ell=1}^L$, as follows. For convenience, we overload Ω^ℓ for the physical layer, and for its corresponding grid.

We continue to let $\mathbf{x}_{p,q} = (x_p, z_q) = (ph, qh)$ in the global indices p and q. The ℓ-th layer is defined via a partition of the z axis, by gathering all q indices such that

$$n_c^\ell < q \leq n_c^{\ell+1},$$

for some increasing sequence n_c^ℓ. The number of grid points in the z direction in layer ℓ is

$$n^\ell = n_c^{\ell+1} - n_c^\ell.$$

(The subscript c stands for cumulative.) Many of the expressions in this Chapter involve local indexing: we let $i = p$ in the x direction, and j such that

$$q = n_c^\ell + j, \qquad 1 \leq j \leq n^\ell,$$

in the z direction.

We hope that there is little risk of confusion in overloading z_j (local indexing) for $z_{n_c^\ell + j}$ (global indexing). Similarly, a quantity like the squared slowness m will be indexed locally in Ω^ℓ via $m_{i,j}^\ell = m_{i,n_c^\ell + j}$. In the sequel, every instance of the superscript ℓ refers to restriction to the ℓ-th layer.

Absorbing boundary conditions in z are then used to complete the definition of the Helmholtz subproblems in each layer Ω^ℓ. Define $\widetilde{\Omega}^\ell$ as the extension of Ω^ℓ in the z direction by n_{pml} points (see Fig. 2-4), and define $\widetilde{m}^\ell$ as the normal extension of m^ℓ given by :

$$\widetilde{m}_{i,j}^\ell = \begin{cases} m_{i,1}^\ell, & j < 0, \\ m_{i,j}^\ell, & 1 \leq j \leq n^\ell, \\ m_{i,n^\ell}^\ell, & j > n^\ell. \end{cases}$$

[5]In Appendix 2.A we use the symmetric formulation as a proof method, and provide the relationship between the Green's functions in the symmetric and unsymmetric cases.

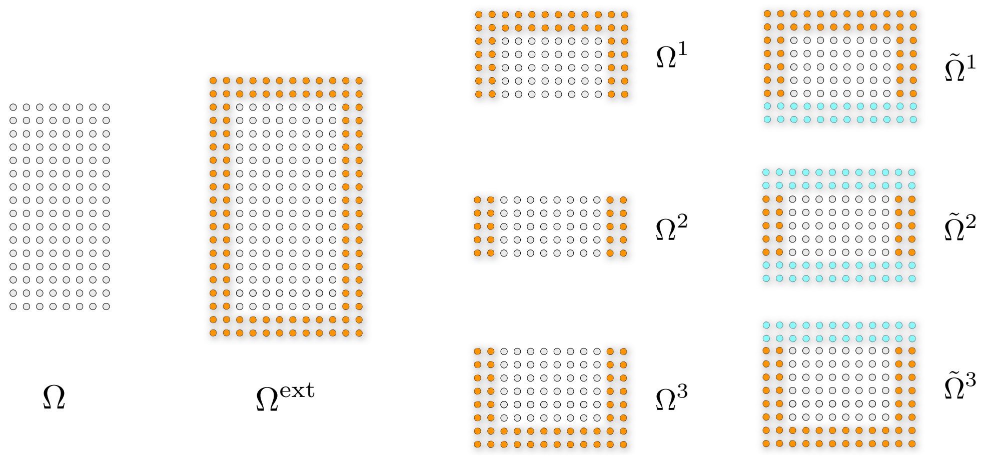


Figure 2-4: The domain Ω is extended to Ω^{ext} by adding the PML nodes (orange). After decomposition into subdomains, the internal boundaries are padded with extra PML nodes (light blue).

The discretization inside Ω^ℓ is otherwise inherited exactly from that of the global problem, via Eq. 2.11. Furthermore, on each PML the discrete differential operator is modified following Eq. 2.13, using the extended local model $\widetilde{m}$. The local problem is denoted as

$$\left(\mathbf{H}^\ell \mathbf{u}^\ell\right)_{i,j} = \mathbf{f}^\ell_{i,j}, \qquad (i,j) \in [\![-n_{\text{pml}}+1, n_x+n_{\text{pml}}]\!] \times [\![-n_{\text{pml}}+1, n^\ell+n_{\text{pml}}]\!], \tag{2.16}$$

where $\mathbf{f}^\ell$ is extended by zero in the PML. The PML's profile is chosen so that the equations for the local and the global problem coincide exactly inside $[\![-n_{\text{pml}}+1, n_x+n_{\text{pml}}]\!] \times [\![1, n^\ell]\!]$.

2.2.3 Discrete Green's Representation Formula

In this section we show that solving the discretized PDE in the volume is equivalent to solving a discrete boundary integral equation from the Green's representation formula, with *local* Green's functions that stem from the local Helmholtz equation. The observation is not surprising in view of very standard results in boundary integral equation theory.

We only gather the results, and refer the reader to Appendix 2.A for the proofs and details of the discrete Green's identities.

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

$$\mathbf{H}^\ell \mathbf{G}^\ell(\mathbf{x}_{i,j}, \mathbf{x}_{i',j'}) = \mathbf{H}^\ell \mathbf{G}^\ell_{i,j,i',j'} = \delta(\mathbf{x}_{i,j} - \mathbf{x}_{i',j'}), \tag{2.17}$$

if $\quad (i,j) \in [\![-n_{\text{pml}}+1, n_x+n_{\text{pml}}]\!] \times [\![-n_{\text{pml}}+1, n^\ell+n_{\text{pml}}]\!]$; where

$$\delta(\mathbf{x}_{i,j} - \mathbf{x}_{i',j'}) = \begin{cases} \frac{1}{h^2}, & \text{if } \mathbf{x}_{i,j} = \mathbf{x}_{i',j'}, \\ 0, & \text{if } \mathbf{x}_{i,j} \neq \mathbf{x}_{i',j'}, \end{cases} \tag{2.18}$$

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

It is notationally more convenient to consider $\mathbf{G}^\ell$ as an operator acting on unknowns at horizontal interfaces, as follows. Again, notations are slighty overloaded.

Definition 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*

$$\left(\mathbf{G}^\ell(z_j, z_k)\mathbf{v}\right)_i = h \sum_{i'=-n_{pml+1}}^{n_x+n_{pml}} \mathbf{G}^\ell((x_i, z_j),(x_{i'}, z_k))\mathbf{v}_{i'}, \tag{2.19}$$

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

Within this context we define the interface identity operator as

$$(\mathbf{Iv})_i = h\mathbf{v}_i. \tag{2.20}$$

As in geophysics, we may refer to z as depth. The layer to layer 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).

Definition 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:*

$$\mathcal{G}_j^{\uparrow,\ell}(\mathbf{v}_{n^\ell}, \mathbf{v}_{n^\ell+1}) = \mathbf{G}^\ell(z_j, z_{n^\ell+1})\left(\frac{\mathbf{v}_{n^\ell+1}-\mathbf{v}_{n^\ell}}{h}\right) - \left(\frac{\mathbf{G}^\ell(z_j, z_{n^\ell+1}) - \mathbf{G}^\ell(z_j, z_{n^\ell})}{h}\right)\mathbf{v}_{n^\ell+1}, \tag{2.21}$$

$$\mathcal{G}_j^{\downarrow,\ell}(\mathbf{v}_0, \mathbf{v}_1) = -\mathbf{G}^\ell(z_j, z_0)\left(\frac{\mathbf{v}_1-\mathbf{v}_0}{h}\right) + \left(\frac{\mathbf{G}^\ell(z_j, z_1) - \mathbf{G}^\ell(z_j, z_0)}{h}\right)\mathbf{v}_0. \tag{2.22}$$

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

The incomplete Green's integrals in Def. 2 use the discrete counterparts of the single and double layer potentials. After some simplifications it is possible to express the incomplete Green's integrals in matrix form:

$$\mathcal{G}_j^{\downarrow,\ell}(\mathbf{v}_0, \mathbf{v}_1) = \frac{1}{h}\left[\begin{array}{cc} \mathbf{G}^\ell(z_j, z_1) & -\mathbf{G}^\ell(z_j, z_0) \end{array}\right] \left(\begin{array}{c} \mathbf{v}_0 \\ \mathbf{v}_1 \end{array}\right), \tag{2.23}$$

$$\mathcal{G}_j^{\uparrow,\ell}(\mathbf{v}_{n^\ell}, \mathbf{v}_{n^\ell+1}) = \frac{1}{h}\left[\begin{array}{cc} -\mathbf{G}^\ell(z_j, z_{n^\ell+1}) & \mathbf{G}^\ell(z_j, z_{n^\ell}) \end{array}\right] \left(\begin{array}{c} \mathbf{v}_{n^\ell} \\ \mathbf{v}_{n^\ell+1} \end{array}\right). \tag{2.24}$$

Definition 3. *Consider the local Newton potential $\mathcal{N}_k^\ell$ applied to a local source $\mathbf{f}^\ell$ as*

$$\mathcal{N}_k^\ell \mathbf{f}^\ell = \sum_{j=1}^{n^\ell} \mathbf{G}^\ell(z_k, z_j)\mathbf{f}_j^\ell. \tag{2.25}$$

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 $1 \leq j \leq n^\ell$.

We can form local solutions to the discrete Helmholtz equation using the local discrete Green's representation formula, as

$$\mathbf{v}^\ell_j = \mathcal{G}^{\uparrow,\ell}_j(\mathbf{v}^\ell_{n^\ell}, \mathbf{v}^\ell_{n^\ell+1}) + \mathcal{G}^{\downarrow,\ell}_j(\mathbf{v}^\ell_0, \mathbf{v}^\ell_1) + \mathcal{N}^\ell_j \mathbf{f}^\ell, \qquad 1 < j < n^\ell. \tag{2.26}$$

This equation is, by construction, a solution to the local problem as long as $1 < j < n^\ell$. It is easy to see that $\mathbf{v}^\ell_j$ is otherwise solution to

$$\mathbf{H}^\ell \mathbf{v}^\ell = \mathbf{f}^\ell + \delta(z_1 - z)\mathbf{v}^\ell_0 - \delta(z_0 - z)\mathbf{v}^\ell_1 - \delta(z_{n^\ell+1} - z)\mathbf{v}^\ell_{n^\ell} + \delta(z_{n^\ell} - z)\mathbf{v}^\ell_{n^\ell+1}. \tag{2.27}$$

Notice that writing local solutions via Eq. 2.26 can be advantageous, since $\mathbf{G}^\ell$ only needs to be stored at interfaces. If a sparse LU factorization of $\mathbf{H}^\ell$ is available, and if the boundary data $\mathbf{v}^\ell_0, \mathbf{v}^\ell_1, \mathbf{v}^\ell_{n^\ell}, \mathbf{v}^\ell_{n^\ell+1}$ are available, then the computation of $\mathbf{v}^\ell$ is reduced to a local sparse solve with the appropriate forcing terms given in Eq. 2.27. This approach can be seen as a generalization of the immersed boundary approach developed by Peskin in [107].

Eq. 2.26 is of particular interest when the data used to build the local solution are the traces of the global solution, as stated in the Lemma below. In other words, Eq. 2.26 continues to hold even when $j = 1$ and $j = n^\ell$.

Lemma 1. *If $\mathbf{u}^\ell_0, \mathbf{u}^\ell_1, \mathbf{u}^\ell_{n^\ell}$, and $\mathbf{u}^\ell_{n^\ell+1}$ are the traces of the global solution at the interfaces, then*

$$\mathbf{u}^\ell_j = \mathcal{G}^{\uparrow,\ell}_j(\mathbf{u}^\ell_{n^\ell}, \mathbf{u}^\ell_{n^\ell+1}) + \mathcal{G}^{\downarrow,\ell}_j(\mathbf{u}^\ell_0, \mathbf{u}^\ell_1) + \mathcal{N}^\ell_j \mathbf{f}^\ell, \tag{2.28}$$

if $1 \leq j \leq n^\ell$, where $\mathbf{u}^\ell$ is the global solution restricted to Ω^ℓ. Moreover,

$$0 = \mathcal{G}^{\uparrow,\ell}_j(\mathbf{u}^\ell_{n^\ell}, \mathbf{u}^\ell_{n^\ell+1}) + \mathcal{G}^{\downarrow,\ell}_j(\mathbf{u}^\ell_0, \mathbf{u}^\ell_1) + \mathcal{N}^\ell_j \mathbf{f}^\ell, \tag{2.29}$$

if $j \leq 0$ and $j \geq n^\ell + 1$.

The proof of Lemma 1 is given in Appendix 2.C. Thus, the problem of solving the global, discrete Helmholtz equation is reduced in an algebraically exact manner to the problem of finding the traces of the solution at the interfaces. Yet, the incomplete Green's operators are not the traditional Schur complements for Ω^ℓ.

What is perhaps most remarkable about those equations is the flexibility that we have in letting $\mathbf{G}^\ell$ be *arbitrary* outside Ω^ℓ – flexiblity that we use by placing a PML.

2.2.4 Discrete Integral Equation

Evaluating the local GRF (Eq. 2.28) at the interfaces $j = 1$ and $j = n^\ell$ of each layer Ω^ℓ results in what we can call a *self-consistency* relation for the solution. Together with a *continuity* condition that the interface data must match at interfaces between layers, we obtain an algebraically equivalent reformulation of the discrete Helmholtz equation.

Definition 4. *Consider the discrete integral formulation for the Helmholtz equation as the system given by*

$$\begin{aligned}
\mathcal{G}_1^{\downarrow,\ell}(\mathbf{u}_0^\ell, \mathbf{u}_1^\ell) + \mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^\ell, \mathbf{u}_{n^\ell+1}^\ell) + \mathcal{N}_1^\ell \mathbf{f}^\ell &= \mathbf{u}_1^\ell, && (2.30)\\
\mathcal{G}_{n^\ell}^{\downarrow,\ell}(\mathbf{u}_0^\ell, \mathbf{u}_1^\ell) + \mathcal{G}_{n^\ell}^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^\ell, \mathbf{u}_{n^\ell+1}^\ell) + \mathcal{N}_{n^\ell}^\ell \mathbf{f}^\ell &= \mathbf{u}_{n^\ell}^\ell, && (2.31)\\
\mathbf{u}_{n^\ell}^\ell &= \mathbf{u}_0^{\ell+1}, && (2.32)\\
\mathbf{u}_{n^\ell+1}^\ell &= \mathbf{u}_1^{\ell+1}, && (2.33)
\end{aligned}$$

if $1 < \ell < L$, *with*

$$\begin{aligned}
\mathcal{G}_{n^1}^{\uparrow,1}(\mathbf{u}_{n^1}^1, \mathbf{u}_{n^\ell+1}^1) + \mathcal{N}_{n^1}^1 \mathbf{f}^1 &= \mathbf{u}_{n^1}^1, && (2.34)\\
\mathbf{u}_{n^1}^1 &= \mathbf{u}_0^2, && (2.35)\\
\mathbf{u}_{n^1+1}^1 &= \mathbf{u}_1^2, && (2.36)
\end{aligned}$$

and

$$\begin{aligned}
\mathcal{G}_1^{\downarrow,L}(\mathbf{u}_0^L, \mathbf{u}_1^L) + \mathcal{N}_{n^L}^L \mathbf{f}^L &= \mathbf{u}_1^L, && (2.37)\\
\mathbf{u}_{n^{L-1}}^{L-1} &= \mathbf{u}_0^L, && (2.38)\\
\mathbf{u}_{n^{L-1}+1}^{L-1} &= \mathbf{u}_1^L. && (2.39)
\end{aligned}$$

Following Def. 2 we can define

$$\underline{\mathbf{M}} = \frac{1}{h}\begin{bmatrix}
-\mathbf{G}_{n,n+1}^1 - \mathbf{I} & \mathbf{G}_{n,n}^1 & 0 & 0 & 0 & 0 \\
\mathbf{G}_{1,1}^2 & -\mathbf{G}_{1,0}^2 - \mathbf{I} & -\mathbf{G}_{1,n+1}^1 & \mathbf{G}_{1,n}^1 & 0 & 0 \\
\mathbf{G}_{n,1}^2 & -\mathbf{G}_{n,0}^2 & -\mathbf{G}_{n,n+1}^2 - \mathbf{I} & \mathbf{G}_{n,n}^2 & 0 & 0 \\
0 & \ddots & \ddots & \ddots & 0 & 0 \\
0 & 0 & \mathbf{G}_{1,1}^{L-1} & -\mathbf{G}_{1,0}^{L-1} - \mathbf{I} & -\mathbf{G}_{1,n+1}^{L-1} & \mathbf{G}_{1,n}^{L-1} \\
0 & 0 & \mathbf{G}_{n,1}^{L-1} & -\mathbf{G}_{n,0}^{L-1} & -\mathbf{G}_{n,n+1}^{L-1} - \mathbf{I} & \mathbf{G}_{n,n}^{L-1} \\
0 & 0 & 0 & 0 & \mathbf{G}_{1,1}^{L} & -\mathbf{G}_{1,0}^{L} - \mathbf{I}
\end{bmatrix}, \quad (2.40)$$

such that the discrete integral system can be written as

$$\underline{\mathbf{M}}\ \underline{\mathbf{u}} = -\begin{pmatrix} \mathcal{N}_{n^1}^1 \mathbf{f}^1 \\ \mathcal{N}_1^2 \mathbf{f}^2 \\ \mathcal{N}_{n^2}^2 \mathbf{f}^2 \\ \vdots \\ \mathcal{N}_1^L \mathbf{f}^L \end{pmatrix} = -\underline{\mathbf{f}}. \quad (2.41)$$

By $\underline{\mathbf{u}}$, we mean the collection of all the interface traces. As a lemma, we recover the required jump condition one grid point past each interface.

Lemma 2. *If* $\mathbf{u}$ *is the solution to the discrete integral system given by Eq. 2.41 then*

$$0 = \mathcal{G}_j^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^\ell, \mathbf{u}_{n^\ell+1}^\ell) + \mathcal{G}_j^{\downarrow,\ell}(\mathbf{u}_0^\ell, \mathbf{u}_1^\ell) + \mathcal{N}_j^\ell \mathbf{f}^\ell, \quad (2.42)$$

if $j = 0$ *or* $j = n^\ell + 1$.

All the algorithms hinge on the following result. We refer the reader to Appendix 2.C for both proofs.

Theorem 1. *The solution of the discrete integral system given by Eq. 2.41 equals the restriction of the solution of the discretized Helmholtz equation given by Eq. 2.15 to the interfaces between subdomains.*

The computational procedure suggested by the reduction to Eq. 2.41 can be decomposed in an offline part given by Alg. 1 and an online part given by Alg. 2.

Algorithm 1. *Offline computation for the discrete integral formulation*

```
 1: function  $H$ = PRECOMPUTATION( $\mathbf{m}, \omega$ )
 2:     for  $\ell = 1 : L$  do
 3:         $\mathbf{m}^\ell = \mathbf{m}\chi_{\Omega^\ell}$                                ▷ partition the model
 4:         $\mathbf{H}^\ell = -\triangle - \mathbf{m}^\ell\omega^2$                       ▷ set local problems
 5:         $[L^\ell, U^\ell] = \mathtt{lu}\left(\mathbf{H}^\ell\right)$                        ▷ factorize local problems
 6:     end for
 7:     for  $\ell = 1 : L$  do                                  ▷ extract Green's functions
 8:         $\mathbf{G}^\ell(z_j, z_{j'}) = (U^\ell)^{-1}(L^\ell)^{-1}\delta(z_{j'})$       ▷ $z_{j'}$ and $z_j$ are in the interfaces
 9:     end for
10:     Form $\underline{\mathbf{M}}$                                        ▷ set up the integral system
11: end function
```

In the offline computation, the domain is decomposed in layers, the local LU factorizations are computed and stored, the local Green's functions are computed on the interfaces by backsubstitution, and the interface-to-interface operators are used to form the discrete integral system. Alg. 1 sets up the data structure and machinery to solve Eq. 2.15 for different right-hand-sides using Alg. 2.

Algorithm 2. *Online computation for the discrete integral formulation*

```
 1: function  $\mathbf{u}$ = HELMHOLTZ SOLVER( $\mathbf{f}$ )
 2:     for  $\ell = 1 : L$  do
 3:         $\mathbf{f}^\ell = \mathbf{f}\chi_{\Omega^\ell}$                                ▷ partition the source
 4:     end for
 5:     for  $\ell = 1 : L$  do
 6:         $\mathcal{N}^\ell\mathbf{f}^\ell = (\mathbf{H}^\ell)^{-1}\mathbf{f}^\ell$                         ▷ solve local problems
 7:     end for
 8:     $\underline{\mathbf{f}} = \left(\mathcal{N}^1_{n^1}\mathbf{f}^1, \mathcal{N}^2_1\mathbf{f}^2, \mathcal{N}^2_{n^2}\mathbf{f}^2, \dots, \mathcal{N}^L_1\mathbf{f}^L\right)^t$   ▷ form r.h.s. for the integral system
 9:     $\underline{\mathbf{u}} = \left(\underline{\mathbf{M}}\right)^{-1}(-\underline{\mathbf{f}})$                   ▷ solve for the traces (Eq. 2.41)
10:     for  $\ell = 1 : L$  do
11:         $\mathbf{u}^\ell_j = \mathcal{G}^{\uparrow,\ell}_j(\mathbf{u}^\ell_{n^\ell}, \mathbf{u}^\ell_{n^\ell+1}) + \mathcal{G}^{\downarrow,\ell}_j(\mathbf{u}^\ell_0, \mathbf{u}^\ell_1) + \mathcal{N}^\ell_j\mathbf{f}^\ell$   ▷ reconstruct local solutions
        (Eq. 2.28)
12:     end for
13:     $\mathbf{u} = (\mathbf{u}^1, \mathbf{u}^2, \dots, \mathbf{u}^{L-1}, \mathbf{u}^L)^t$            ▷ concatenate the local solutions
14: end function
```

The matrix $\underline{\mathbf{M}}$ that results from the discrete integral equations[6] is block sparse and tightly block banded, so the discrete integral system can in principle be factorized by a block LU algorithm. However, and even though the integral system (after pre-computation of the local Green's functions) represents a reduction of one dimension in the Helmholtz problem, solving the integral formulation directly can be extremely hard to parallelize and prohibitively expensive for large systems. The only feasible option in such cases is to use iterative methods; however, the condition number of $\underline{\mathbf{M}}$ is $\mathcal{O}(h^{-2})$, resulting in a high number of iterations for convergence.

The rest of this Chapter is devoted to the question of designing a good preconditioner for $\underline{\mathbf{M}}$ in regimes of propagating waves, which makes an iterative solver competitive for the discrete integral system.

Remark 1. *The idea behind the discrete integral system in Def. 4 is to take the limit to the interfaces by approaching them from the interior of each layer. It is possible to write an equivalent system by taking the limit from outside. By Lemma 2 the wavefield is zero outside the slab, hence by taking this limit we can write the equation for* $\mathbf{u}$ *as*

$$\underline{\mathbf{M}}_{\mathbf{0}}\mathbf{u} = -\begin{pmatrix} \mathcal{N}^1_{n^1+1}\mathbf{f}^1 \\ \mathcal{N}^2_0\mathbf{f}^2 \\ \mathcal{N}^2_{n^2+1}\mathbf{f}^2 \\ \vdots \\ \mathcal{N}^L_0\mathbf{f}^L \end{pmatrix} = -\underline{\mathbf{f}}_{\mathbf{0}}, \tag{2.43}$$

where

$$\underline{\mathbf{M}}_{\mathbf{0}} = \frac{1}{h}\begin{bmatrix} -\mathbf{G}^1_{n+1,n+1} & \mathbf{G}^1_{n+1,n} & 0 & 0 & 0 & 0 \\ \mathbf{G}^2_{0,1} & -\mathbf{G}^2_{0,0} & -\mathbf{G}^1_{0,n+1} & \mathbf{G}^1_{0,n} & 0 & 0 \\ \mathbf{G}^2_{n+1,1} & -\mathbf{G}^2_{n+1,0} & -\mathbf{G}^2_{n+1,n+1} & \mathbf{G}^2_{n+1,n} & 0 & 0 \\ 0 & \ddots & \ddots & \ddots & 0 & 0 \\ 0 & 0 & \mathbf{G}^{L-1}_{0,1} & -\mathbf{G}^{L-1}_{0,0} & -\mathbf{G}^{L-1}_{0,n+1} & \mathbf{G}^{L-1}_{0,n} \\ 0 & 0 & \mathbf{G}^{L-1}_{n+1,1} & -\mathbf{G}^{L-1}_{n+1,0} & -\mathbf{G}^{L-1}_{n+1,n+1} & \mathbf{G}^{L-1}_{n+1,n} \\ 0 & 0 & 0 & 0 & \mathbf{G}^L_{0,1} & -\mathbf{G}^L_{0,0} \end{bmatrix}. \tag{2.44}$$

The systems given by Eq. 2.41 and Eq. 2.43 are equivalent; however, the first one has more intuitive implications hence it is the one we deal with it in the immediate sequel.

2.3 Polarization

Interesting structure is revealed when we reformulate the system given by Eq. 2.41, by splitting each interface field into two "polarized traces". We start by describing the proper discrete expression of the concept seen in the introduction.

[6]The resulting integral system can be seen as the discrete counterpart of a surface integral equation (see [147] and references therein) in which we used numerical Green's functions instead of analytical ones.

2.3.1 Polarized Wavefields

Definition 5. *(Polarization via annihilation relations.) A wavefield* $\mathbf{u}$ *is said to be up-going at the interface* $\Gamma_{\ell,\ell+1}$ *if it satisfies the annihilation relation given by*

$$\mathcal{G}_1^{\downarrow,\ell+1}(\mathbf{u}_0, \mathbf{u}_1) = 0. \tag{2.45}$$

We denote it as $\mathbf{u}^{\uparrow}$*. Analogously, a wavefield* $\mathbf{u}$ *is said to be down-going at the interface* $\Gamma_{\ell,\ell+1}$ *if it satisfies the annihilation relation given by*

$$\mathcal{G}_{n^\ell}^{\uparrow,\ell}(\mathbf{u}_{n^\ell}, \mathbf{u}_{n^\ell+1}) = 0. \tag{2.46}$$

We denote it as $\mathbf{u}^{\downarrow}$*.*

A pair $(\mathbf{u}_0^{\uparrow}, \mathbf{u}_1^{\uparrow})$ satisfies the up-going annihilation relation at $\Gamma_{\ell,\ell+1}$ when it is a wavefield radiated from below $\Gamma_{\ell,\ell+1}$.

Up and down arrows are convenient notations, but it should be remembered that the polarized fields contain locally reflected waves in addition to transmitted waves, hence are not purely directional as in a uniform medium. The quality of polarization is directly inherited from the choice of local Green's function in the incomplete Green's operators $\mathcal{G}^{\downarrow,\ell+1}$ and $\mathcal{G}^{\uparrow,\ell}$. In the polarization context, we may refer to these operators as *annihilators.*

The main feature of a polarized wave is that it can be extrapolated outside the domain using only the Dirichlet data at the boundary. In particular, we can extrapolate one grid point using the extrapolator in Def. 6.

Definition 6. *Let* $\mathbf{v}_1$ *be the trace of a wavefield at* $j = 1$ *in local coordinates. Then define the up-going one-sided extrapolator as*

$$\mathcal{E}_{\ell,\ell+1}^{\uparrow}\mathbf{v}_1 = \left(\mathbf{G}^{\ell+1}(z_1, z_1)\right)^{-1}\mathbf{G}^{\ell+1}(z_1, z_0)\mathbf{v}_1. \tag{2.47}$$

Analogously, for a trace $\mathbf{v}_{n^\ell}$ *at* $j = n^\ell$*, define the down-going one-sided extrapolator as*

$$\mathcal{E}_{\ell,\ell+1}^{\downarrow}\mathbf{v}_{n^\ell} = \left(\mathbf{G}^{\ell}(z_{n^\ell}, z_{n^\ell})\right)^{-1}\mathbf{G}^{\ell}(z_{n^\ell}, z_{n^\ell+1})\mathbf{v}_{n^\ell}. \tag{2.48}$$

Extrapolators reproduce the polarized waves.

Lemma 3. *Let* $\mathbf{u}^{\uparrow}$ *be an up-going wavefield. Then* $\mathbf{u}_0^{\uparrow}$ *is completely determined by* $\mathbf{u}_1^{\uparrow}$*, as*

$$\mathbf{u}_0^{\uparrow} = \mathcal{E}_{\ell,\ell+1}^{\uparrow}\mathbf{u}_1^{\uparrow}. \tag{2.49}$$

Analogously for a down-going wave $\mathbf{u}^{\downarrow}$*, we have*

$$\mathbf{u}_{n^\ell+1}^{\downarrow} = \mathcal{E}_{\ell,\ell+1}^{\downarrow}\mathbf{u}_{n^\ell}^{\downarrow}. \tag{2.50}$$

Lemma 4. *The extrapolator satisfies the following properties:*

$$\mathbf{G}^{\ell+1}(z_0, z_j) = \mathcal{E}_{\ell,\ell+1}^{\uparrow}\mathbf{G}^{\ell+1}(z_1, z_j), \qquad \textit{for } j \geq 1, \tag{2.51}$$

and

$$\mathbf{G}^{\ell}(z_{n^\ell+1}, z_j) = \mathcal{E}^{\downarrow}_{\ell,\ell+1}\mathbf{G}^{\ell}(z_{n^\ell}, z_j), \qquad \textit{for } j \leq n^{\ell}. \tag{2.52}$$

Moreover, we have the jump conditions

$$\mathbf{G}^{\ell+1}(z_0, z_0) - h\mathcal{E}^{\uparrow}_{\ell,\ell+1} = \mathcal{E}^{\uparrow}_{\ell,\ell+1}\mathbf{G}^{\ell+1}(z_1, z_0), \tag{2.53}$$

and

$$\mathbf{G}^{\ell}(z_{n^\ell+1}, z_{n^\ell+1}) - h\mathcal{E}^{\downarrow}_{\ell,\ell+1} = \mathcal{E}^{\downarrow}_{\ell,\ell+1}\mathbf{G}^{\ell}(z_{n^\ell}, z_{n^\ell+1}). \tag{2.54}$$

In one dimension, the proof of Lemma 4 (in Appendix 2.C) is a direct application of the nullity theorem [132] and the cofactor formula, in which $\mathcal{E}^{\uparrow}_{j,j+1}$ is the ratio between two co-linear vectors. In two dimensions, the proof is slightly more complex but follows the same reasoning.

Lemma 5. *If* $\mathbf{u}^{\uparrow}$ *is an up-going wave-field, then the annihilation relation holds inside the layer, i.e.*

$$\mathcal{G}^{\downarrow,\ell+1}_{j}(\mathbf{u}^{\uparrow}_0, \mathbf{u}^{\uparrow}_1) = 0, \qquad \textit{for } j \geq 1. \tag{2.55}$$

Analogously, if $\mathbf{u}^{\uparrow}$ *is a down-going wave-field, then the annihilation relation holds inside the layer, i.e.*

$$\mathcal{G}^{\uparrow,\ell}_{j}(\mathbf{u}^{\downarrow}_{n^\ell}, \mathbf{u}^{\downarrow}_{n^\ell}) = 0, \qquad \textit{for } j \leq n^{\ell}. \tag{2.56}$$

Remark 2. *We gather from the proof of Lemma 4 that we can define extrapolators inside the domain as well. In fact, from Proposition 4 in Appendix 2.B, we can easily show that*

$$\left[\mathbf{G}^{\ell}(z_j, z_j)\right]^{-1}\mathbf{G}^{\ell}(z_j, z_{j+1})\mathbf{G}^{\ell}(z_j, z_k) = \mathbf{G}^{\ell}(z_{j+1}, z_k) \qquad \textit{for } k \leq j, \tag{2.57}$$

and

$$\left[\mathbf{G}^{\ell}(z_{j+1}, z_{j+1})\right]^{-1}\mathbf{G}^{\ell}(z_{j+1}, z_j)\mathbf{G}^{\ell}(z_{j+1}, z_k) = \mathbf{G}^{\ell}(z_j, z_k) \qquad \textit{for } j \leq k. \tag{2.58}$$

2.3.2 Polarized Traces

Some advantageous cancellations occur when doubling the number of unknowns, and formulating a system on traces of polarized wavefields at interfaces. In this approach, each trace is written as the sum of two polarized components. We define the collection of *polarized traces* as

$$\underline{\underline{\mathbf{u}}} = \begin{pmatrix} \underline{\mathbf{u}}^{\downarrow} \\ \underline{\mathbf{u}}^{\uparrow} \end{pmatrix}, \tag{2.59}$$

such that $\underline{\mathbf{u}} = \underline{\mathbf{u}}^{\uparrow} + \underline{\mathbf{u}}^{\downarrow}$. As previously, we underline symbols to denote collections of traces. We now underline twice to denote polarized traces. The original system gives rise to the equations

$$\begin{bmatrix} \underline{\mathbf{M}} & \underline{\mathbf{M}} \end{bmatrix} \underline{\underline{\mathbf{u}}} = -\underline{\mathbf{f}}. \tag{2.60}$$

This polarized formulation increases the number of unknowns; however, the number of equations remains the same. Some further knowledge should be used to obtain a square system. We present three mathematically equivalent, though algorithmically different approaches to closing the system:

1. a first approach is to impose the annihilation polarization conditions;
2. a second approach is to impose the extrapolator relations in tandem with the annihilation conditions; and
3. a third approach uses the jump conditions to further simplify expressions.

The second formulation yields an efficient preconditioner consisting of decoupled block-triangular matrices. However, its inversion relies on inverting some of the dense diagonal blocks, which is problematic without extra knowledge about these blocks. The third formulation is analogous to the second one, but altogether alleviates the need for dense block inversion. It is this last formulation that we benchmark in the numerical section. We prefer to present all three versions in a sequence, for clarity of the presentation.

2.3.3 Annihilation relations

We can easily close the system given by Eq. 2.60 by explicitly imposing that the out-going traces at each interface satisfies the annihilation conditions (Eq. 2.63 and Eq. 2.64). We then obtain a system of equations given by :

$$\mathcal{G}_1^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\uparrow},\mathbf{u}_1^{\ell,\uparrow}) + \mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow},\mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_1^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow},\mathbf{u}_1^{\ell,\downarrow}) + \mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\downarrow},\mathbf{u}_{n^\ell+1}^{\ell,\downarrow}) + \mathcal{N}_1^{\ell}\mathbf{f}^\ell = \mathbf{u}_1^{\ell,\uparrow} + \mathbf{u}_1^{\ell,\downarrow}, \tag{2.61}$$

$$\mathcal{G}_{n^\ell}^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\uparrow},\mathbf{u}_1^{\ell,\uparrow}) + \mathcal{G}_{n^\ell}^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow},\mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_{n^\ell}^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow},\mathbf{u}_1^{\ell,\downarrow}) + \mathcal{G}_{n^\ell}^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\downarrow},\mathbf{u}_{n^\ell+1}^{\ell,\downarrow}) + \mathcal{N}_{n^\ell}^{\ell}\mathbf{f}^\ell = \mathbf{u}_{n^\ell}^{\ell,\uparrow} + \mathbf{u}_{n^\ell}^{\ell,\downarrow}, \tag{2.62}$$

$$\mathcal{G}_{n^\ell}^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\downarrow},\mathbf{u}_{n^\ell+1}^{\ell,\downarrow}) = 0, \tag{2.63}$$

$$\mathcal{G}_1^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\uparrow},\mathbf{u}_1^{\ell,\uparrow}) = 0. \tag{2.64}$$

plus the continuity conditions. To obtain the matrix form of this system, we define the global annihilator matrices by

$$\underline{\mathbf{A}}^{\downarrow} = \frac{1}{h}\left[\begin{array}{cccccc} -\mathbf{G}^1_{n,n+1} & \mathbf{G}^1_{n,n} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\mathbf{G}^2_{n,n+1} & \mathbf{G}^2_{n,n} & 0 & 0 \\ 0 & 0 & 0 & \ddots & \ddots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\mathbf{G}^{L-1}_{n,n+1} & \mathbf{G}^{L-1}_{n,n} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right], \tag{2.65}$$

and

$$\underline{\mathbf{A}}^{\uparrow} = \frac{1}{h}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{G}^2_{1,1} & -\mathbf{G}^2_{1,0} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \\ 0 & \ddots & \ddots & \ddots & 0 & 0 \\ 0 & 0 & \mathbf{G}^{L-1}_{1,1} & -\mathbf{G}^{L-1}_{1,0} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{G}^{L}_{1,1} & -\mathbf{G}^{L}_{1,0} \end{bmatrix}. \tag{2.66}$$

Definition 7. *We define the polarized system completed with annihilator conditions as*

$$\begin{bmatrix} \underline{\mathbf{M}} & \underline{\mathbf{M}} \\ \underline{\mathbf{A}}^{\downarrow} & \underline{\mathbf{A}}^{\uparrow} \end{bmatrix} \underline{\underline{\mathbf{u}}} = \begin{pmatrix} -\underline{\mathbf{f}} \\ 0 \end{pmatrix}. \tag{2.67}$$

By construction, if $\underline{\underline{\mathbf{u}}}$ is solution to Eq. 2.67, then $\underline{\mathbf{u}} = \underline{\mathbf{u}}^{\uparrow} + \underline{\mathbf{u}}^{\downarrow}$ is solution to Eq. 2.41. Moreover, the non-zero blocks of $\underline{\mathbf{A}}^{\uparrow}$ and $\underline{\mathbf{A}}^{\downarrow}$ are full rank, and given their nested structure it follows that $[\underline{\mathbf{A}}^{\downarrow} \;\; \underline{\mathbf{A}}^{\uparrow}]$ is full row rank as well.

2.3.4 Extrapolation conditions

One standard procedure for preconditioning a system such as Eq. 2.41 is to use a block Jacobi preconditioner, or a block Gauss-Seidel preconditioner. Several solvers based on domain decomposition use the latter as a preconditioner, and typically call it a multiplicative Schwarz iteration. In our case however, once the system is augmented from $\underline{\mathbf{M}}$ to $[\underline{\mathbf{M}} \;\; \underline{\mathbf{M}}]$, and completed, it loses its banded structure. The proper form can be restored in the system of Def. 7 by a sequence of steps that we now outline.

It is clear from Def. 5 that some of the terms of $\underline{\mathbf{M}}$ contain the annihilation relations. Those terms should be subtracted from the relevant rows of $\underline{\mathbf{M}}$, resulting in new submatrices $\underline{\mathbf{M}}^{\downarrow}$ and $\underline{\mathbf{M}}^{\uparrow}$. Completion of the system is advantageously done in a different way, by encoding polarization via the extrapolator conditions from Def. 6, rather than the annihiliation conditions. The system given by Eq. 2.67 is then equivalent to

$$\mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}^{\ell,\uparrow}_{n^\ell}, \mathbf{u}^{\ell,\uparrow}_{n^\ell+1}) + \mathcal{G}_1^{\downarrow,\ell}(\mathbf{u}^{\ell,\downarrow}_0, \mathbf{u}^{\ell,\downarrow}_1) + \mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}^{\ell,\downarrow}_{n^\ell}, \mathbf{u}^{\ell,\downarrow}_{n^\ell+1}) + \mathcal{N}_1^{\ell}\mathbf{f}^{\ell} = \mathbf{u}^{\ell,\uparrow}_1 + \mathbf{u}^{\ell,\downarrow}_1, \tag{2.68}$$

$$\mathcal{G}_{n^\ell}^{\downarrow,\ell}(\mathbf{u}^{\ell,\uparrow}_0, \mathbf{u}^{\ell,\uparrow}_1) + \mathcal{G}_{n^\ell}^{\uparrow,\ell}(\mathbf{u}^{\ell,\uparrow}_{n^\ell}, \mathbf{u}^{\ell,\uparrow}_{n^\ell+1}) + \mathcal{G}_{n^\ell}^{\downarrow,\ell}(\mathbf{u}^{\ell,\downarrow}_0, \mathbf{u}^{\ell,\downarrow}_1) + \mathcal{N}_{n^\ell}^{\ell}\mathbf{f}^{\ell} = \mathbf{u}^{\ell,\uparrow}_{n^\ell} + \mathbf{u}^{\ell,\downarrow}_{n^\ell}, \tag{2.69}$$

$$\mathbf{u}^{\ell,\downarrow}_{n^\ell+1} = \mathcal{E}^{\downarrow}_{\ell,\ell+1}(\mathbf{u}^{\ell,\downarrow}_{n^\ell}), \tag{2.70}$$

$$\mathbf{u}^{\ell+1,\uparrow}_0 = \mathcal{E}^{\uparrow}_{\ell,\ell+1}(\mathbf{u}^{\ell+1,\uparrow}_1). \tag{2.71}$$

We can switch to a matrix form of these equations, by letting

$$\underline{\mathbf{M}}^{\downarrow} = \frac{1}{h}\begin{bmatrix} -\mathbf{I} & 0 & 0 & 0 & 0 & 0 \\ \mathbf{G}^2_{1,1} & -\mathbf{G}^2_{1,0} - \mathbf{I} & -\mathbf{G}^1_{1,n+1} & \mathbf{G}^1_{1,n} & 0 & 0 \\ \mathbf{G}^2_{n,1} & -\mathbf{G}^2_{n,0} & -\mathbf{I} & 0 & 0 & 0 \\ \\ 0 & \ddots & \ddots & \ddots & 0 & 0 \\ 0 & 0 & \mathbf{G}^{L-1}_{1,1} & -\mathbf{G}^{L-1}_{1,0} - \mathbf{I} & -\mathbf{G}^{L-1}_{1,n+1} & \mathbf{G}^{L-1}_{1,n} \\ 0 & 0 & \mathbf{G}^{L-1}_{n,1} & -\mathbf{G}^{L-1}_{n,0} & -\mathbf{I} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{G}^{L}_{1,1} & -\mathbf{G}^{L}_{1,0} - \mathbf{I} \end{bmatrix}, \tag{2.72}$$

$$\underline{\mathbf{M}}^{\uparrow} = \frac{1}{h}\begin{bmatrix} -\mathbf{G}^1_{n,n+1} - \mathbf{I} & \mathbf{G}^1_{n,n} & 0 & 0 & 0 & 0 \\ 0 & -\mathbf{I} & -\mathbf{G}^1_{1,n+1} & \mathbf{G}^1_{1,n} & 0 & 0 \\ \mathbf{G}^2_{n,1} & -\mathbf{G}^2_{n,0} & -\mathbf{G}^2_{n,n+1} - \mathbf{I} & \mathbf{G}^2_{n,n} & 0 & 0 \\ 0 & \ddots & \ddots & \ddots & 0 & 0 \\ 0 & 0 & 0 & -\mathbf{I} & -\mathbf{G}^{L-1}_{1,n+1} & \mathbf{G}^{L-1}_{1,n} \\ 0 & 0 & \mathbf{G}^{L-1}_{n,1} & -\mathbf{G}^{L-1}_{n,0} & -\mathbf{G}^{L-1}_{n,n+1} - \mathbf{I} & \mathbf{G}^{L-1}_{n,n} \\ 0 & 0 & 0 & 0 & 0 & -\mathbf{I} \end{bmatrix}, \tag{2.73}$$

$$\underline{\mathbf{E}}^{\downarrow} = \frac{1}{h}\begin{bmatrix} h\left(\mathbf{G}^1_{n,n}\right)^{-1}\mathbf{G}^1_{n,n+1} & -\mathbf{I} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h\left(\mathbf{G}^2_{n,n}\right)^{-1}\mathbf{G}^2_{n,n+1} & -\mathbf{I} & 0 & 0 \\ 0 & 0 & 0 & \ddots & \ddots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & h\left(\mathbf{G}^{L-1}_{n,n}\right)^{-1}\mathbf{G}^{L-1}_{n,n+1} & -\mathbf{I} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{2.74}$$

$$\underline{\mathbf{E}}^{\uparrow} = \frac{1}{h}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ -\mathbf{I} & h\left(\mathbf{G}^2_{1,1}\right)^{-1}\mathbf{G}^2_{1,0} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \ddots & \ddots & \ddots & 0 & 0 \\ 0 & 0 & -\mathbf{I} & h\left(\mathbf{G}^{L-1}_{1,1}\right)^{-1}\mathbf{G}^{L-1}_{1,0} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\mathbf{I} & h\left(\mathbf{G}^{L}_{1,1}\right)^{-1}\mathbf{G}^{L}_{1,0} \end{bmatrix}. \tag{2.75}$$

The sparsity pattern of the system formed by these block matrices is given by Fig. 2-5 *(left)*. We arrive at the following definition.

Definition 8. *We define the polarized system completed with the extrapolation relations as*

$$\begin{bmatrix} \underline{\mathbf{M}}^{\downarrow} & \underline{\mathbf{M}}^{\uparrow} \\ \underline{\mathbf{E}}^{\downarrow} & \underline{\mathbf{E}}^{\uparrow} \end{bmatrix} \underline{\underline{\mathbf{u}}} = \begin{pmatrix} -\underline{\mathbf{f}} \\ 0 \end{pmatrix}. \tag{2.76}$$

The interesting feature of this system, in contrast to the previous formulation, is that $\mathbf{u}^{\ell,\downarrow}_{n^\ell}$ is undisturbed (multiplied by an identity block) both in $\underline{\mathbf{M}}^{\downarrow}$ and in $\underline{\mathbf{E}}^{\downarrow}$. Similarly, $\mathbf{u}^{\ell,\uparrow}_1$ is left undisturbed by $\underline{\mathbf{M}}^{\uparrow}$ and $\underline{\mathbf{E}}^{\uparrow}$. This is apparent from Fig. 2-5 *(left)*. Following this observation we can permute the rows of the matrix to obtain

$$\left(\begin{bmatrix} \underline{\mathbf{D}}^{\downarrow} & 0 \\ 0 & \underline{\mathbf{D}}^{\uparrow} \end{bmatrix} + \begin{bmatrix} 0 & \underline{\mathbf{U}} \\ \underline{\mathbf{L}} & 0 \end{bmatrix}\right) \underline{\underline{\mathbf{u}}} = \underline{\mathbf{P}} \begin{pmatrix} -\underline{\mathbf{f}} \\ 0 \end{pmatrix}, \tag{2.77}$$

where the diagonal blocks $\underline{\mathbf{D}}^{\downarrow}$ and $\underline{\mathbf{D}}^{\downarrow}$ are respectively upper triangular and lower triangular, with identity blocks on the diagonal; $\underline{\mathbf{P}}$ is an appropriate 'permutation' matrix; and $\underline{\mathbf{U}}$ and $\underline{\mathbf{L}}$ are block sparse matrices. We define the matrices

$$\underline{\mathbf{D}}^{\text{extrap}} = \begin{bmatrix} \underline{\mathbf{D}}^{\downarrow} & 0 \\ 0 & \underline{\mathbf{D}}^{\uparrow} \end{bmatrix}, \qquad \underline{\mathbf{R}}^{\text{extrap}} = \begin{bmatrix} 0 & \underline{\mathbf{U}} \\ \underline{\mathbf{L}} & 0 \end{bmatrix}. \tag{2.78}$$

We can observe the sparsity pattern of the permuted system in Fig. 2-5 *(right)*.

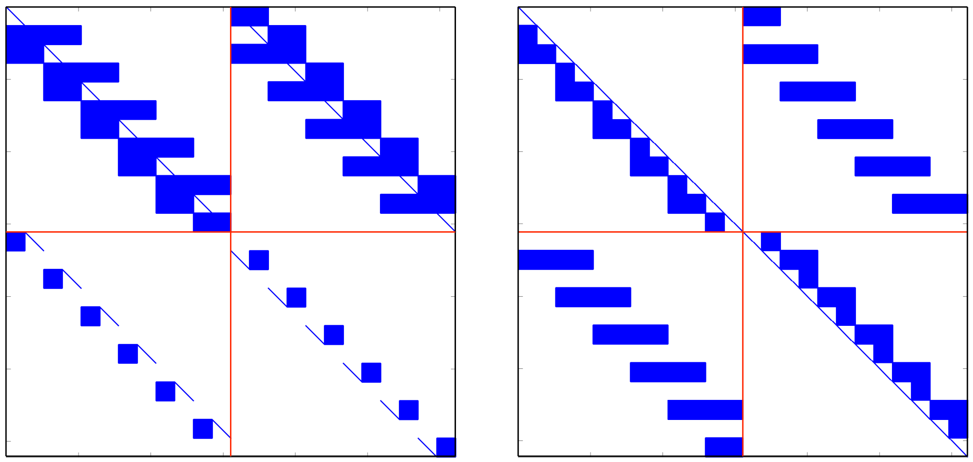


Figure 2-5: Left: Sparsity pattern of the system in Eq. 2.76. Right: Sparsity pattern of reordered system in Eq. 2.76.

2.3.5 Jump condition

The polarized system in Def. 8 can be easily preconditioned using a block Jacobi iteration for the 2×2 block matrix, which yields a procedure to solve the Helmholtz equation. Moreover, using $\underline{\mathbf{D}}^{\text{extrap}}$ as a pre-conditioner within GMRES, yields remarkable results. However, in order to obtain the desired structure, we need to use the extrapolator, whose construction involves inverting small dense blocks. This can be costly and inefficient when dealing with large interfaces (or surfaces in 3D). In order to avoid the inversion of any local operator we exploit the properties of the discrete GRF to obtain an equivalent system that avoids any unnecesary dense linear algebra operation.

Following Remark 1, we can complete the polarized system by imposing,

$$\left[\begin{array}{cc} \underline{\mathbf{M}}_{\mathbf{0}} & \underline{\mathbf{M}}_{\mathbf{0}} \end{array}\right] \underline{\underline{\mathbf{u}}} = -\underline{\mathbf{f}}_{\mathbf{0}}, \tag{2.79}$$

where $\underline{\mathbf{M}}_{\mathbf{0}}$ encodes the jump condition for the GRF. However, these blocks do not preserve $\mathbf{u}^{\uparrow}$ and $\mathbf{u}^{\downarrow}$ like $\underline{\mathbf{D}}^{\text{extrap}}$ did, because $\underline{\mathbf{M}}_{\mathbf{0}}$ does not have identities on the diagonal. Fortunately, it is possible to include the information contained in the annihilation relations directly into $\underline{\mathbf{M}}_{\mathbf{0}}$, exactly as we did with $\underline{\mathbf{M}}$. Lemma 6 summarizes the expression resulting from incorporating the extrapolation conditions into $\underline{\mathbf{M}}_{\mathbf{0}}$.

Lemma 6. *If* $\underline{\underline{\mathbf{u}}}$ *is solution to the system given by Def. 8 then*

$$\mathbf{u}_0^{\uparrow,\ell} = \mathcal{E}^{\uparrow}_{\ell,\ell+1}\mathbf{u}_1^{\uparrow,\ell} = \mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}^{\ell,\uparrow}_{n^\ell}, \mathbf{u}^{\ell,\uparrow}_{n^\ell+1}) + \mathcal{G}_0^{\downarrow,\ell}(\mathbf{u}^{\ell,\downarrow}_0, \mathbf{u}^{\ell,\downarrow}_1) + \mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}^{\ell,\downarrow}_{n^\ell}, \mathbf{u}^{\ell,\downarrow}_{n^\ell+1}) + \mathcal{N}_0^{\ell}\mathbf{f}^{\ell}, \tag{2.80}$$

$$\mathbf{u}_{n^\ell+1}^{\downarrow,\ell} = \mathcal{E}^{\downarrow}_{\ell,\ell+1}\mathbf{u}_{n^\ell}^{\downarrow,\ell} = \mathcal{G}_{n^\ell+1}^{\downarrow,\ell}(\mathbf{u}^{\ell,\uparrow}_0, \mathbf{u}^{\ell,\uparrow}_1) + \mathcal{G}_{n^\ell+1}^{\uparrow,\ell}(\mathbf{u}^{\ell,\uparrow}_{n^\ell}, \mathbf{u}^{\ell,\uparrow}_{n^\ell+1}) + \mathcal{G}_{n^\ell+1}^{\downarrow,\ell}(\mathbf{u}^{\ell,\downarrow}_0, \mathbf{u}^{\ell,\downarrow}_1) + \mathcal{N}_{n^\ell+1}^{\ell}\mathbf{f}^{\ell}. \tag{2.81}$$

The jump conditions in Lemma 2 are heavily used in the proof, see the Appendix.

We now replace the extrapolation relations by Eq. 2.80 and Eq. 2.81, which leads to the next system :

$$\mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow},\mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_1^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow},\mathbf{u}_1^{\ell,\downarrow}) + \mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\downarrow},\mathbf{u}_{n^\ell+1}^{\ell,\downarrow}) + \mathcal{N}_1^\ell \mathbf{f}^\ell = \mathbf{u}_1^{\ell,\uparrow} + \mathbf{u}_1^{\ell,\downarrow}, \tag{2.82}$$

$$\mathcal{G}_{n^\ell}^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\uparrow},\mathbf{u}_1^{\ell,\uparrow}) + \mathcal{G}_{n^\ell}^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow},\mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_{n^\ell}^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow},\mathbf{u}_1^{\ell,\downarrow}) + \mathcal{N}_{n^\ell}^\ell \mathbf{f}^\ell = \mathbf{u}_{n^\ell}^{\ell,\uparrow} + \mathbf{u}_{n^\ell}^{\ell,\downarrow}, \tag{2.83}$$

$$\mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow},\mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_0^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow},\mathbf{u}_1^{\ell,\downarrow}) + \mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\downarrow},\mathbf{u}_{n^\ell+1}^{\ell,\downarrow}) + \mathcal{N}_0^\ell \mathbf{f}^\ell = \mathbf{u}_0^{\ell,\uparrow}, \tag{2.84}$$

$$\mathcal{G}_{n^\ell+1}^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\uparrow},\mathbf{u}_1^{\ell,\uparrow}) + \mathcal{G}_{n^\ell+1}^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow},\mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_{n^\ell+1}^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow},\mathbf{u}_1^{\ell,\downarrow}) + \mathcal{N}_{n^\ell+1}^\ell \mathbf{f}^\ell = \mathbf{u}_{n^\ell}^{\ell,\downarrow}. \tag{2.85}$$

We define the matrix form of Eq. 2.84 and Eq. 2.85 by

$$\underline{\mathbf{M}}_0^\uparrow = \frac{1}{h}\begin{bmatrix}
-\mathbf{G}_{n+1,n+1}^1 & \mathbf{G}_{n+1,n}^1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-\mathbf{I} & 0 & -\mathbf{G}_{0,n+1}^2 & \mathbf{G}_{0,n}^2 & 0 & 0 & 0 & 0 \\
\mathbf{G}_{n+1,1}^2 & -\mathbf{G}_{n+1,0}^2 & -\mathbf{G}_{n+1,n+1}^2 & \mathbf{G}_{n+1,n}^2 & 0 & 0 & 0 & 0 \\
0 & 0 & -\mathbf{I} & 0 & -\mathbf{G}_{0,n+1}^3 & \mathbf{G}_{0,n}^3 & 0 & 0 \\
0 & 0 & \mathbf{G}_{n+1,1}^3 & -\mathbf{G}_{n+1,0}^3 & -\mathbf{G}_{n+1,n+1}^3 & \mathbf{G}_{n+1,n}^3 & 0 & 0 \\
0 & 0 & 0 & \ddots & \ddots & \ddots & \ddots & 0 \\
0 & 0 & 0 & 0 & -\mathbf{I} & 0 & -\mathbf{G}_{0,n+1}^{L-1} & \mathbf{G}_{0,n}^{L-1} \\
0 & 0 & 0 & 0 & \mathbf{G}_{n+1,1}^{L-1} & -\mathbf{G}_{n+1,0}^{L-1} & -\mathbf{G}_{n+1,n+1}^{L-1} & \mathbf{G}_{n+1,n}^{L-1} \\
0 & 0 & 0 & 0 & 0 & 0 & -\mathbf{I} & 0
\end{bmatrix},$$

$$\underline{\mathbf{M}}_0^\downarrow = \frac{1}{h}\begin{bmatrix}
0 & -\mathbf{I} & 0 & 0 & 0 & 0 & 0 & 0 \\
\mathbf{G}_{0,1}^2 & -\mathbf{G}_{0,0}^2 & -\mathbf{G}_{0,n+1}^2 & \mathbf{G}_{0,n}^2 & 0 & 0 & 0 & 0 \\
\mathbf{G}_{n+1,1}^2 & -\mathbf{G}_{n+1,0}^2 & 0 & -\mathbf{I} & 0 & 0 & 0 & 0 \\
0 & 0 & \mathbf{G}_{0,1}^3 & -\mathbf{G}_{0,0}^3 & -\mathbf{G}_{0,n+1}^3 & \mathbf{G}_{0,n}^3 & 0 & 0 \\
0 & 0 & \mathbf{G}_{n+1,1}^3 & -\mathbf{G}_{n+1,0}^3 & 0 & -\mathbf{I} & 0 & 0 \\
0 & 0 & 0 & \ddots & \ddots & \ddots & \ddots & 0 \\
0 & 0 & 0 & 0 & \mathbf{G}_{0,1}^{L-1} & -\mathbf{G}_{0,0}^{L-1} - \mathbf{G}_{0,n+1}^{L-1} & \mathbf{G}_{0,n}^{L-1} & \\
0 & 0 & 0 & 0 & \mathbf{G}_{n+1,1}^{L-1} & -\mathbf{G}_{n+1,0}^{L-1} & 0 & -\mathbf{I} \\
0 & 0 & 0 & 0 & 0 & 0 & \mathbf{G}_{0,1}^L & -\mathbf{G}_{0,0}^L
\end{bmatrix},$$

and

$$\underline{\mathbf{f}}_0 = \begin{pmatrix} \mathcal{N}_{n^1+1}^1 \mathbf{f}^1 \\ \mathcal{N}_0^2 \mathbf{f}^2 \\ \mathcal{N}_{n^2+1}^2 \mathbf{f}^2 \\ \vdots \\ \mathcal{N}_0^L \mathbf{f}^L \end{pmatrix}. \tag{2.86}$$

The resulting matrix equations are as follows.

Definition 9. *We define the polarized system completed with jump conditions as*

$$\underline{\underline{\mathbf{M}}}\,\underline{\underline{\mathbf{u}}} = \begin{bmatrix} \underline{\mathbf{M}}^\downarrow & \underline{\mathbf{M}}^\uparrow \\ \underline{\mathbf{M}_\mathbf{0}}^\downarrow & \underline{\mathbf{M}_\mathbf{0}}^\uparrow \end{bmatrix} \underline{\underline{\mathbf{u}}} = \begin{pmatrix} -\underline{\mathbf{f}} \\ -\underline{\mathbf{f}_\mathbf{0}} \end{pmatrix}. \tag{2.87}$$

By construction, the system given by Eq. 2.87 has identities at the same locations as Eq. 2.76. The same row permutation $\underline{\mathbf{P}}$ as before will result in triangular diagonal blocks with identity blocks on the diagonal:

$$\underline{\mathbf{P}} \begin{bmatrix} \underline{\mathbf{M}}^{\downarrow} & \underline{\mathbf{M}}^{\uparrow} \\ \underline{\mathbf{M}_\mathbf{0}}^{\downarrow} & \underline{\mathbf{M}_\mathbf{0}}^{\uparrow} \end{bmatrix} \underline{\underline{\mathbf{u}}} = \left(\underline{\mathbf{D}}^{\text{jump}} + \underline{\mathbf{R}}^{\text{jump}}\right) \underline{\underline{\mathbf{u}}} = \underline{\mathbf{P}} \begin{pmatrix} -\underline{\mathbf{f}} \\ -\underline{\mathbf{f}_\mathbf{0}} \end{pmatrix}, \tag{2.88}$$

where,

$$\underline{\mathbf{D}}^{\text{jump}} = \begin{bmatrix} \underline{\mathbf{D}}^{\downarrow,\text{jump}} & 0 \\ 0 & \underline{\mathbf{D}}^{\uparrow,\text{jump}} \end{bmatrix}, \qquad \underline{\mathbf{R}}^{\text{jump}} = \begin{bmatrix} 0 & \underline{\mathbf{U}}^{\text{jump}} \\ \underline{\mathbf{L}}^{\text{jump}} & 0 \end{bmatrix}. \tag{2.89}$$

We can observe the sparsity pattern in Fig 2-6 *(right)*.

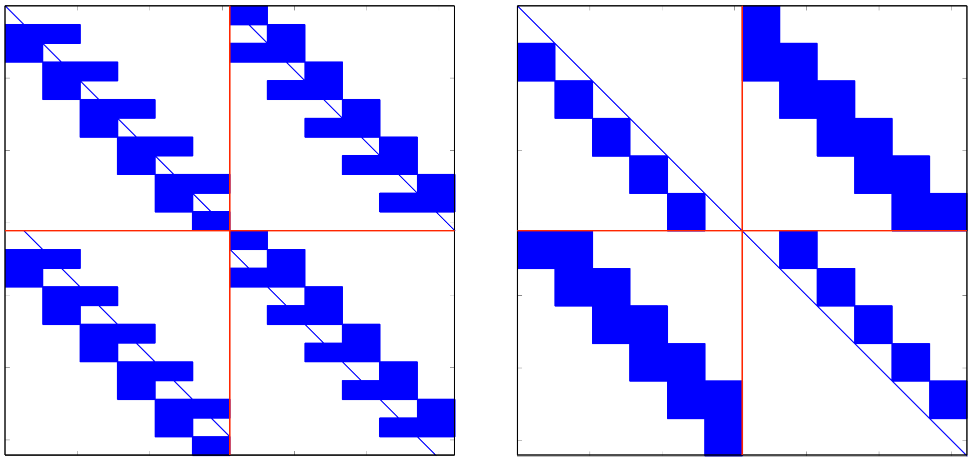


Figure 2-6: Left: Sparsity pattern of the system in Eq. 2.87. Right: Sparsity pattern of the permuted system in Eq. 2.89.

For reference, here is the explicit form of the blocks of $\underline{\mathbf{D}}^{\text{jump}}$.

$$\underline{\mathbf{D}}^{\downarrow,\text{jump}} = \frac{1}{h} \begin{bmatrix} -\mathbf{I} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\mathbf{I} & 0 & 0 & 0 & 0 & 0 \\ \mathbf{G}^2_{n,1} & -\mathbf{G}^2_{n,0} & -\mathbf{I} & 0 & 0 & 0 & 0 \\ \mathbf{G}^2_{n+1,1} & -\mathbf{G}^2_{n+1,0} & 0 & -\mathbf{I} & 0 & 0 & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & 0 & 0 \\ 0 & 0 & 0 & \mathbf{G}^{L-1}_{n,1} & -\mathbf{G}^{L-1}_{n,0} & -\mathbf{I} & 0 \\ 0 & 0 & 0 & \mathbf{G}^{L-1}_{n+1,1} & -\mathbf{G}^{L-1}_{n+1,0} & 0 & -\mathbf{I} \end{bmatrix}, \tag{2.90}$$

$$\underline{\mathbf{D}}^{\uparrow,\text{jump}} = \frac{1}{h} \begin{bmatrix} -\mathbf{I} & 0 & -\mathbf{G}^2_{0,n+1} & \mathbf{G}^2_{0,n} & 0 & 0 & 0 \\ 0 & -\mathbf{I} & -\mathbf{G}^2_{1,n+1} & \mathbf{G}^2_{1,n} & 0 & 0 & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & -\mathbf{I} & 0 & -\mathbf{G}^{L-1}_{0,n+1} & \mathbf{G}^{L-1}_{0,n} \\ 0 & 0 & 0 & 0 & -\mathbf{I} & -\mathbf{G}^{L-1}_{1,n+1} & \mathbf{G}^{L-1}_{1,n} \\ 0 & 0 & 0 & 0 & 0 & -\mathbf{I} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\mathbf{I} \end{bmatrix}, \tag{2.91}$$

$$\underline{\mathbf{L}}^{\text{jump}} = \frac{1}{h} \begin{bmatrix} \mathbf{G}^2_{0,1} & -\mathbf{G}^2_{0,0} & -\mathbf{G}^2_{0,n+1} & \mathbf{G}^2_{0,n} & 0 & 0 \\ \mathbf{G}^2_{1,1} & -\mathbf{G}^2_{1,0} - \mathbf{I} & -\mathbf{G}^2_{1,n+1} & \mathbf{G}^2_{1,n} & 0 & 0 \\ 0 & \ddots & \ddots & \ddots & 0 & 0 \\ 0 & 0 & \mathbf{G}^{L-1}_{0,1} & -\mathbf{G}^{L-1}_{0,0} & -\mathbf{G}^{L-1}_{0,n+1} & \mathbf{G}^{L-1}_{0,n} \\ 0 & 0 & \mathbf{G}^{L-1}_{1,1} & -\mathbf{G}^{L-1}_{1,0} - \mathbf{I} & -\mathbf{G}^{L-1}_{1,n+1} & \mathbf{G}^{L-1}_{1,n} \\ 0 & 0 & 0 & 0 & \mathbf{G}^{L}_{0,1} & -\mathbf{G}^{L}_{0,0} \\ 0 & 0 & 0 & 0 & \mathbf{G}^{L}_{1,1} & -\mathbf{G}^{L}_{1,0} - \mathbf{I} \end{bmatrix}, \tag{2.92}$$

and

$$\underline{\mathbf{U}}^{\text{jump}} = \frac{1}{h} \begin{bmatrix} -\mathbf{G}^1_{n,n+1} - \mathbf{I} & \mathbf{G}^1_{n,n} & 0 & 0 & 0 & 0 \\ -\mathbf{G}^1_{n+1,n+1} & \mathbf{G}^1_{n+1,n} & 0 & 0 & 0 & 0 \\ \mathbf{G}^2_{n,1} & -\mathbf{G}^2_{n,0} - \mathbf{I} & -\mathbf{G}^2_{n,n+1} & \mathbf{G}^2_{n,n} & 0 & 0 \\ \mathbf{G}^2_{n+1,1} & -\mathbf{G}^2_{n+1,0} & -\mathbf{G}^2_{n+1,n+1} & \mathbf{G}^2_{n+1,n} & 0 & 0 \\ 0 & \ddots & \ddots & \ddots & 0 & 0 \\ 0 & 0 & \mathbf{G}^{L-1}_{n,1} & -\mathbf{G}^{L-1}_{n,0} - \mathbf{I} & -\mathbf{G}^{L-1}_{n,n+1} & \mathbf{G}^{L-1}_{n+1,n} \\ 0 & 0 & \mathbf{G}^{L-1}_{n+1,1} & -\mathbf{G}^{L-1}_{n+1,0} & -\mathbf{G}^{L-1}_{n+1,n+1} & \mathbf{G}^{L-1}_{n+1,n} \end{bmatrix}. \tag{2.93}$$

The formulations given by Eq. 2.76 and Eq. 2.87 are equivalent. The following lemma can be proved from Lemma 2 and Lemma 6.

Proposition 1. *$\underline{\underline{\mathbf{u}}}$ is solution to the system in Def. 8 if and only if $\underline{\underline{\mathbf{u}}}$ is solution to the system in Def. 9.*

The numerical claims of this Chapter concern the system in Eq. 2.88, and preconditioned with the direct inversion of $\underline{\mathbf{D}}^{\text{jump}}$ defined by Eq. 2.89.

2.4 Preconditioners

2.4.1 Jacobi Iteration

In this section we let $\underline{\mathbf{D}}$ for either $\underline{\mathbf{D}}^{\text{jump}}$ or $\underline{\mathbf{D}}^{\text{extrap}}$. While we mostly use the jump formulation in practice, the structure of the preconditioner is common to both formulations.

Inverting any such $\underline{\mathbf{D}}$ is trivial using block back-substitution for each block $\underline{\mathbf{D}}^{\downarrow}$, $\underline{\mathbf{D}}^{\uparrow}$, because they have a triangular structure, and their diagonals consist of identity blocks. Physically, the inversion of $\underline{\mathbf{D}}$ results in two sweeps of the domain (top-down and bottom-up) for computing transmitted waves from incomplete Green's formulas. This procedure is close enough to Gauss-Seidel to be referred to as such[7].

Algorithm 3. *Jacobi iteration*

```
1: function  $\mathbf{u}$ = JACOBI( $\mathbf{f}, \epsilon_{tol}$ )
2:     $\underline{\mathbf{u}}^0 = (\mathbf{u}^{\downarrow}, \mathbf{u}^{\uparrow})^t = 0$
3:     while  $\|\underline{\mathbf{u}}^{n+1} - \underline{\mathbf{u}}^n\| / \|\underline{\mathbf{u}}^n\| > \epsilon_{tol}$  do
4:         $\underline{\mathbf{u}}^{n+1} = (\underline{\mathbf{D}})^{-1}(\underline{\mathbf{P}}\, \underline{\widetilde{\mathbf{f}}} - \underline{\mathbf{R}}\, \underline{\mathbf{u}}^n)$
5:     end while
6:     $\mathbf{u} = \mathbf{u}^{\uparrow,n} + \mathbf{u}^{\downarrow,n}$
```

[7]Another possibility would be to call it a multiplicative Schwarz iteration.

```
7: end function
```

Alg. 3 is generic: the matrices $\underline{\mathbf{D}}$ and $\underline{\mathbf{R}}$ can either arise from Def. 8 or Def. 9; and $\widetilde{\underline{\mathbf{f}}}$ can either be $(\underline{\mathbf{f}}, 0)^t$ or $(\underline{\mathbf{f}}, \underline{\mathbf{f}}_0)^t$ depending on the system being solved.

2.4.2 GMRES

Alg. 3 is primarily an iterative solver for the discrete integral system given by Def. 4, and can be seen as only using the polarized system given by Eq. 2.87 in an auxiliary fashion. Unfortunately, the number of iterations needed for Alg. 3 to converge to a given tolerance often increases as a fractional power of the number of sub-domains. We address this problem by solving Eq. 2.87 in its own right, using GMRES combined with Alg. 3 as a preconditioner. As we illustrate in the sequel, the resulting number of iterations is now roughly constant in the number of subdomains. The preconditioner is defined as follows.

Algorithm 4. *Block-Jacobi Preconditioner*

```
1: function $\underline{\mathbf{u}}$ = PRECONDITIONER($\widetilde{\underline{\mathbf{f}}}, n_{it}$)
2:     $\underline{\mathbf{u}}^0 = (\mathbf{u}^{\downarrow}, \mathbf{u}^{\uparrow})^t = 0$
3:     for $n = 0$, $n < n_{it}$, $n++$ do
4:         $\underline{\mathbf{u}}^{n+1} = (\underline{\mathbf{D}})^{-1}(\underline{\mathbf{P}}\,\widetilde{\underline{\mathbf{f}}} - \underline{\mathbf{R}}\,\underline{\mathbf{u}}^n)$
5:     end for
6:     $\underline{\mathbf{u}} = \underline{\mathbf{u}}^{n_{it}}$
7: end function
```

If we suppose that $n_{\text{it}} = 1$ (good choices are $n_{\text{it}} = 1$ or 2), then the convergence of preconditioned GMRES will depend on the clustering of the eigenvalues of

$$(\underline{\mathbf{D}})^{-1}\,\underline{\mathbf{P}}\,\underline{\underline{\mathbf{M}}} = I + (\underline{\mathbf{D}})^{-1}\underline{\mathbf{R}} = \begin{bmatrix} I & (\underline{\mathbf{D}}^{\downarrow})^{-1}\underline{\mathbf{U}} \\ (\underline{\mathbf{D}}^{\uparrow})^{-1}\underline{\mathbf{L}} & I \end{bmatrix}. \tag{2.94}$$

We can compute these eigenvalues from the zeros of the characteristic polynomial. Using a well-known property of Schur complements, we get

$$\det((\underline{\mathbf{D}})^{-1}\,\underline{\mathbf{P}}\,\underline{\underline{\mathbf{M}}} - \lambda \underline{I}) = \det(I - \lambda I)\det\left(I - \lambda I - \left((\underline{\mathbf{D}}^{\uparrow})^{-1}\underline{\mathbf{L}}(I-\lambda I)^{-1}(\underline{\mathbf{D}}^{\downarrow})^{-1}\underline{\mathbf{U}}\right)\right). \tag{2.95}$$

This factorization means that half of the eigenvalues are exactly one. For the remaining half, we write the characteristic polynomial as

$$\det\left((I-\lambda I)^2 - \left((\underline{\mathbf{D}}^{\uparrow})^{-1}\underline{\mathbf{L}}(\underline{\mathbf{D}}^{\downarrow})^{-1}\underline{\mathbf{U}}\right)\right). \tag{2.96}$$

Let $\mu = (1-\lambda)^2$, and consider the new eigenvalue problem

$$\det\left(\mu I - \left((\underline{\mathbf{D}}^{\uparrow})^{-1}\underline{\mathbf{L}}\,(\underline{\mathbf{D}}^{\downarrow})^{-1}\underline{\mathbf{U}}\right)\right) = 0. \tag{2.97}$$

Hence the eigenvalues of $(\underline{\mathbf{D}})^{-1}\,\underline{\mathbf{P}}\,\underline{\underline{\mathbf{M}}}$ are given by 1 and $1 \pm \sqrt{\mu_i}$, where μ_i are the eigenvalues of $(\underline{\mathbf{D}}^{\uparrow})^{-1}\underline{\mathbf{L}}\,(\underline{\mathbf{D}}^{\downarrow})^{-1}\underline{\mathbf{U}}$, and $\pm\sqrt{\mu_i}$ is either complex square root of μ_i.

Notice that $\pm\sqrt{\mu_i}$ coincide with the eigenvalues of the iteration matrix $(\underline{\mathbf{D}})^{-1}\underline{\mathbf{R}}$ of the Gauss-Seidel method.

The smaller the bulk of the $|\mu_i|$, the more clustered $1\pm\sqrt{\mu_i}$ around 1, the faster the convergence of preconditioned GMRES. This intuitive notion of clustering is robust to the presence of a small number of outliers with large $|\mu_i|$. The spectral radius $\rho\left((\underline{\mathbf{D}})^{-1}\underline{\mathbf{R}}\right) = \max_i \sqrt{|\mu_i|}$, however, is not robust to outlying μ_i, hence our remark about preconditioned GMRES being superior to Gauss-Seidel. These outliers do occur in practice in heterogeneous media; see Fig. 2-8.

To give a physical interpretation to the eigenvalues μ_i of $(\underline{\mathbf{D}}^{\uparrow})^{-1}\underline{\mathbf{L}}\,(\underline{\mathbf{D}}^{\downarrow})^{-1}\underline{\mathbf{U}}$, consider the role of each block:

- $\underline{\mathbf{U}}$ maps $\mathbf{u}^{\uparrow}$ to $\mathbf{u}^{\downarrow}$ traces within a layer; it takes into account all the *reflections* and other scattering phenomena that turn waves from up-going to down-going inside the layer. Vice-versa for $\underline{\mathbf{L}}$, which maps down-going to up-going traces.

- $(\underline{\mathbf{D}}^{\downarrow})^{-1}$ maps down-going traces at interfaces to down-going traces at all the other layers below it; it is a *transmission* of down-going waves in a top-down sweep of the domain. Vice-versa for $(\underline{\mathbf{D}}^{\uparrow})^{-1}$, which transmits up-going waves in a bottom-up sweep. It is easy to check that transmission is done via the computation of incomplete discrete GRF.

Hence the combination $(\underline{\mathbf{D}}^{\uparrow})^{-1}\underline{\mathbf{L}}\,(\underline{\mathbf{D}}^{\downarrow})^{-1}\underline{\mathbf{U}}$ generates reflected waves from the heterogeneities within a layer, propagates them down to every other layer below it, reflects them again from the heterogeneities within each of those layers, and propagates the result back up through the domain. The magnitude of the succession of these operations is akin to a coefficient of “double reflection" accounting for scattering through the whole domain. We therefore expect the size of the eigenvalues $|\mu_i|$ to be proportional to the strength of the medium heterogeneities, including how far the PML are to implementing absorbing boundary conditions. The numerical exeriments support this interpretation.

As an example, Fig. 2-7 shows the eigenvalues of $(\underline{\mathbf{D}})^{-1}\;\underline{\mathbf{P}}\;\underline{\underline{\mathbf{M}}}$ when the media is homogeneous, but with a PML of varying quality. Fig 2-8 shows the eigenvalues of $(\underline{\mathbf{D}})^{-1}\;\underline{\mathbf{P}}\;\underline{\underline{\mathbf{M}}}$ in a rough medium, and again with PML of varying quality.

We can expect that, as long as the medium is free of resonant cavities, the eigenvalues of the preconditioned matrix will cluster around 1, implying that GMRES will converge fast. Assuming that the reflection coefficients depend weakly on the frequency, we can expect that the performance of the preconditioner will deteriorate no more than marginally as the frequency increases.

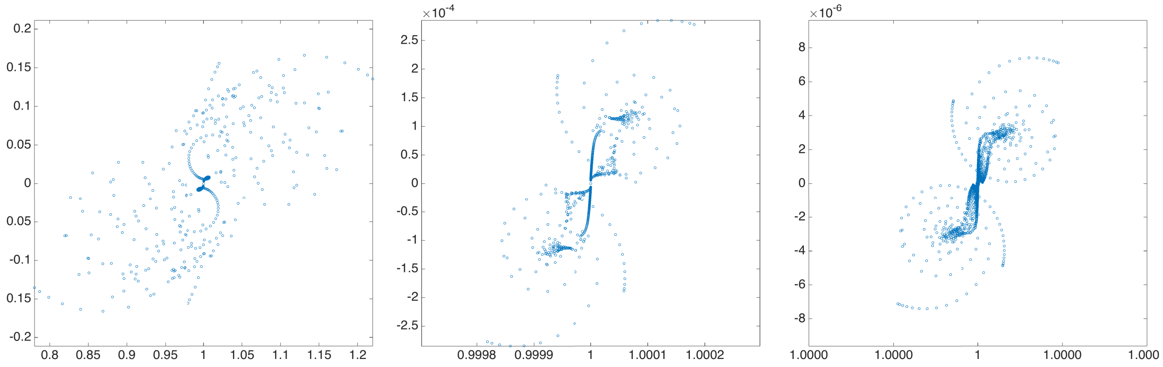


Figure 2-7: Eigenvalues in the complex plane (the abscissa and ordinate present the real and imaginary part respectively) of preconditioned boundary system, for a homogeneous media , $L = 3$, $n =$, $\omega = 30$; and 5 (*left*), 30 (*center*) and 100 (*right*) PML points. Notice the scale of the axes.

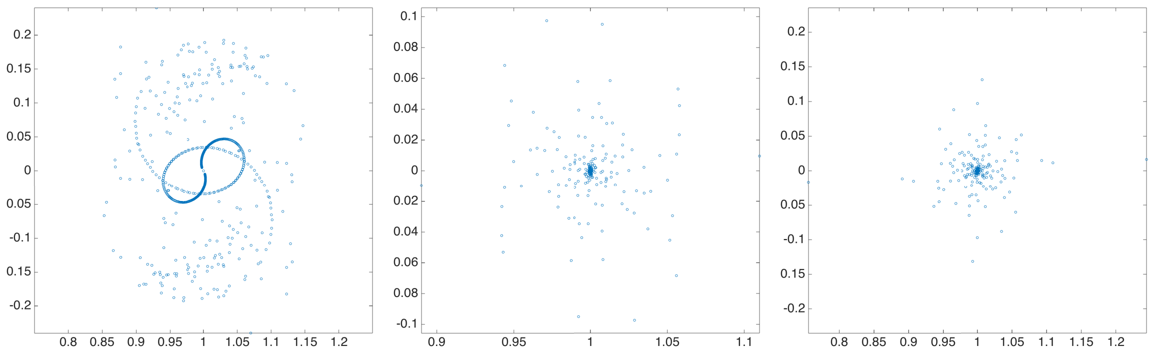


Figure 2-8: Eigenvalues in the complex plane (the abscissa and ordinate present the real and imaginary part respectively) of the preconditioned boundary system for the Marmousi2 model (Fig. 2-12), $L = 3$, $n = 300$, $\omega = 30$; and 5 (*left*), 30 (*center*) and 100 (*right*) PML points.

2.5 Partitioned low-rank matrices

Let $n \sim \sqrt{N}$ for the number of points per dimension, and L be the number of subdomains. So far, the complexity of solving the discrete integral system in polarized form, assuming a constant number of preconditioned GMRES iterations, is dominated by the application of $\underline{\mathbf{D}}^{-1}$ and $\underline{\mathbf{R}}$ in the update step $\underline{\mathbf{u}}^{n+1} = (\underline{\mathbf{D}})^{-1}(\underline{\mathbf{P}}\ \underline{\widetilde{\mathbf{f}}} - \underline{\mathbf{R}}\ \underline{\mathbf{u}}^{n})$. Each of the $\mathcal{O}(L)$ nonzero blocks of $\underline{\mathbf{D}}^{-1}$ and $\underline{\mathbf{R}}$ is n-by-n. A constant number of applications of these matrices therefore results in a complexity that scales as $\mathcal{O}(n^2L)$. This scaling is at best linear in the number N of volume unknowns.

It is the availability of fast algorithms for $\underline{\mathbf{D}}^{-1}$ and $\underline{\mathbf{R}}$, i.e., for the blocks of $\underline{\underline{\mathbf{M}}}$, that can potentially lower the complexity of solving Eq. 2.41 down to sublinear in N. In this setting, the best achievable complexity would be $\mathcal{O}(nL)$ – the complexity of specifying $\mathcal{O}(L)$ traces of size $\mathcal{O}(n)$. As mentioned earlier, the overall online complexity can be sublinear in N if we parallelize the operations of reading off $\mathbf{f}$ in the volume, and forming the volume unknowns $\mathbf{u}$ from the traces.

We opt for what is perhaps the simplest and best-known algorithm for fast application of arbitrary kernels in discrete form: an adaptive low-rank partitioning of the matrix. This choice is neither original nor optimal in the high-frequency regime, but it gives rise to elementary code. More sophisticated approaches have been proposed elsewhere, including by one of us in [28], but the extension of those ideas to the kernel-independent framework is not immediate. This section is therefore added for the sake of algorithmic completeness, and for clarification of the ranks and complexity scalings that arise from low-rank partitioning in the high-frequency regime.

2.5.1 Compression

The blocks of $\underline{\underline{\mathbf{M}}}$, which stem from the discretization of interface-to-interface operators, are compressed using the recursive Alg. 5. The result of this algorithm is a quadtree structure on the original matrix, where the leaves are maximally large square submatrices with fixed ϵ-rank. We follow [18] in calling this structure partitioned low-rank (PLR). An early reference for PLR matrices is the work of Jones, Ma, and Rokhlin in 1994 [86]. PLR matrices are a special case of $\mathcal{H}$-matrices[8].

It is known [9], that the blocks of matrices such as $\underline{\underline{\mathbf{M}}}$ can have low rank, provided they obey an admissibility condition that takes into account the distance between blocks. In regimes of high frequencies and rough heterogeneous media, this admissibility condition becomes more stringent in ways that are not entirely understood yet. See [28, 52] for partial progress.

Neither of the usual non-adaptive admissibility criteria seems adequate in our case. The "nearby interaction" blocks (from one interface to itself) have a singularity along the diagonal, but tend to otherwise have large low-rank blocks in media close to uniform [96]. The "remote interaction" blocks (from one interface to the next) do not have the diagonal problem, but have a wave vector diversity that generates higher ranks. Adaptivity is therefore a very natural choice, and is not a problem in situations where the only operation of interest is the matrix-vector product.

For a fixed accuracy ϵ and a fixed rank $r_{\max}$, we say that a partition is admissible if every block has ϵ-rank less or equal than $r_{\max}$. Alg. 5 finds the smallest admissible partition within the quadtree generated by recursive dyadic partitioning of the indices.

Algorithm 5. *Partitioned Low Rank matrix*

```
1: function H = PLR($M$, $r_{max}$, $\epsilon$)
2:     $[U, \Sigma, V] = \mathbf{svds}(M, r_{max}+1)$
3:     if $\Sigma(r_{max}+1, r_{max}+1) < \epsilon$ then
4:         H.data = {$U \cdot \Sigma$, $V^t$}
5:         H.id = 'c'                                   ▷ leaf node
6:     else
7:         $M = \begin{bmatrix} M_{1,1} & M_{1,2} \\ M_{2,1} & M_{2,2} \end{bmatrix}$                  ▷ block partitioning
```

[8]We reserve the term $\mathcal{H}$-matrix for hierarchical structures preserved by algebraic operations like multiplication and inversion, like in [9].

```
8:          for $i$ = 1:2 do
9:              for $j$ = 1:2 do
10:                 H.data{i,j} = PLR($M_{i,j}$, $r_{max}$, $\epsilon$ )
11:             end for
12:         end for
13:         H.id = 'h'                                        ▷ branch node
14:     end if
15: end function
```

Fig. 2-9 depicts the hierarchical representation of a PLR matrix of the compressed matrix for the nearby interactions *(left)* and the remote interactions *(right)*. Once the matrices are compressed in PLR form, we can easily define a fast matrix-vector multiplication using Alg. 6.

Algorithm 6. *PLR-vector Multiplication*

```
1: function Y = MATVEC($x$)
2:     if H.id == 'c' then                                    ▷ If leaf node
3:         $y$ = H.data{1}·(H.data{2}·x)        ▷ perform mat-vec using SVD factors
4:     else                                                   ▷ If branch node
5:         for $i$ = 1:2 do                                   ▷ recurse over children
6:             $y_1$+=matvec(H.data{i,1},$x_{1:end/2}$)
7:             $y_2$+=matvec(H.data{i,2},$x_{end/2:end}$)
8:         end for
9:         $y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$            ▷ concatenate solution from recursion
10:    end if
11: end function
```

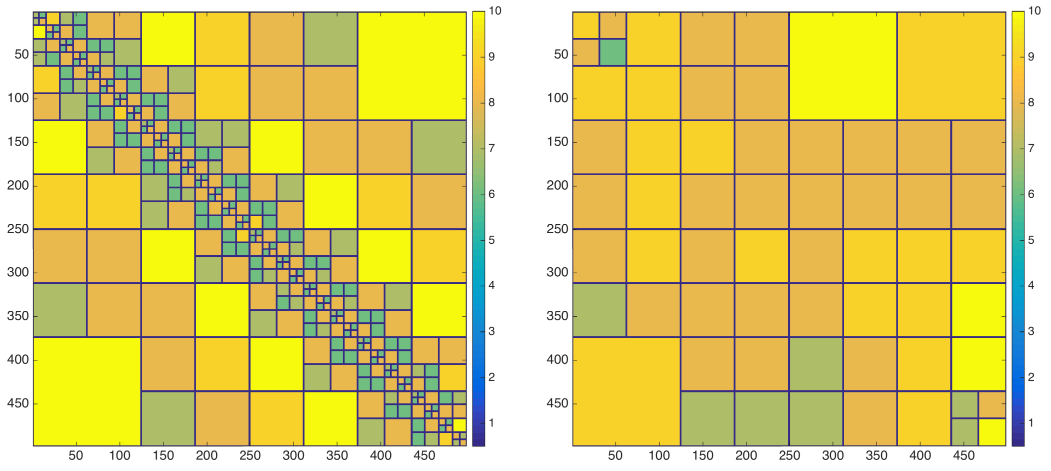


Figure 2-9: Illustration of compressed Green's matrices in PLR form (ϵ-ranks ≤ 10, $\epsilon = 10^{-9}$). Each color represents a different numerical rank. Left: nearby interactions. Right: remote interactions.

Alg. 6 yields a fast matrix vector multiplication; however, given its recursive nature the constant for the scaling can become large. This phenomenon is overwhelming if Alg. 6 is implemented in a scripting language such as MATLAB or Python. In the case of compiled languages, recursions tend to not be correctly optimized by the compiler, increasing the constant in front of the asymptotic complexity scaling. To reduce the constants, the PLR matrix-vector multiplication was implemented via a sparse factorization as illustrated by Fig 2-10. This factorization allows us to take advantage of highly optimized sparse multiplication routines. An extra advantage of using such routines is data locality and optimized cache management when performing several matrix-vector multiplications at the same time, which can be performed as a matrix-matrix multiplication. This kind of sparse factorization is by no means new; we suggest as a reference Section 4 in [1].

The maximum local rank of the compression scheme can be seen as a tuning parameter. If it is too small, it will induce small off-diagonal blocks, hindering compression and deteriorating the complexity of the matrix-vector multiplication. On the other hand, a large maximum rank will induce a partition with big dense diagonal blocks that should be further compressed, resulting in the same adverse consequences.

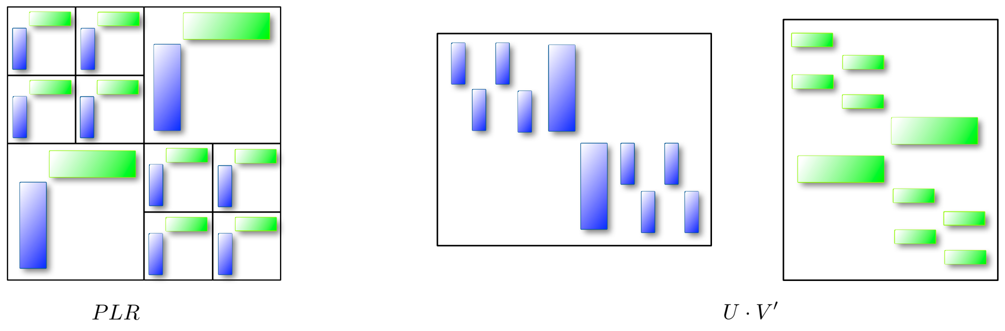


Figure 2-10: Illustration of the sparse form of a PLR matrix. Left: PLR matrix. Right: its sparse factorization form.

2.5.2 Compression scalings

It is difficult to analyze the compressibility of Green's functions without access to an explicit formula for their kernel. In this section we make the assumption of smooth media and single-valued traveltimes, for which a geometrical optics approximation such as

$$G(\mathbf{x},\mathbf{y};\omega) \simeq a_\omega(\mathbf{x},\mathbf{y})e^{i\omega\tau(\mathbf{x},\mathbf{y})}, \tag{2.98}$$

holds. Here $\tau(\mathbf{x},\mathbf{y})$ solves an eikonal equation; and $a_\omega(\mathbf{x},\mathbf{y})$ is an amplitude factor, smooth except at $\mathbf{x} = \mathbf{y}$, and with a minor[9] dependence on ω. Assume that both a

[9]In the standard geometrical optics asymptotic expansion, $a \sim \sum_{j\geq 0} a_j\omega^{-j}$ is polyhomogeneous with increasingly negative orders in ω.

and τ are $C^{\infty}_{\mathbf{x},\mathbf{y}}$ away from $x = y$, with smoothness constants bounded independently of ω.

The following result is a straightforward generalization of a result in [28, 52] and would be proved in much the same way.

Lemma 7. *(High-frequency admissibility condition) Consider G as in Eq. 2.98, with $\mathbf{x} \in A$ and $\mathbf{y} \in B$ where A and B are two rectangles. Let d_A, d_B be the respective diameters of A and B. If*

$$d_A d_B \leq \frac{dist(A, B)}{\omega},$$

then the ϵ-rank of the restriction of G to $A \times B$ is, for some $R_\epsilon > 0$, bounded by R_ϵ (independent of ω).

The scaling is tight given the geometry of the problem. When dealing with 1D geometries and uniform media as in [96] we can observe better scalings. For a more comprehensive analysis of the different scalings depending on the geometry see [61].

In our case, we refer to “nearby interactions" as the case when $\mathbf{x}$ and $\mathbf{y}$ are on the same interface (horizontal edge), and “remote interactions" when $\mathbf{x}$ and $\mathbf{y}$ are on opposite horizontal edges of a layer Ω^{ℓ}. In both cases, A and B are 1D segments, but the geometry is 2D for the remote interactions. Our partitioning scheme limits A and B to have the same length $d_A = d_B$, hence the lemma implies that low ranks can in general only occur provided

$$d_A \leq C \frac{1}{\sqrt{\omega}}.$$

In other words, d_A is at best proportional to the square root of the (representative) spatial wavelength, and even smaller when A and B are close. A square block of the discrete $\mathbf{G}$ with $d_A \sim \frac{1}{\sqrt{\omega}}$, on a grid with n points per dimension, would be of size $\sim \frac{n}{\sqrt{\omega}} \times \frac{n}{\sqrt{\omega}}$. We call such a block representative; it suffices to understand the complexity scalings under the assumption that all blocks are representative[10].

Let $\omega \sim n^{\rho}$ for some ρ such as $\frac{1}{2}$ or 1. This implies that the representative block of $\mathbf{G}$ has size $n^{1-\rho/2} \times n^{1-\rho/2}$ (where we drop the constants for simplicity). Given that the ϵ-rank is bounded by R_ϵ, this block can be compressed using Alg 5 in two matrices of size $n^{1-\rho/2} \times R_\epsilon$ and $R_\epsilon \times n^{1-\rho/2}$. We have $\mathcal{O}(n^{\rho})$ such blocks in $\mathbf{G}$.

We can easily compute an estimate for the compression ratio; we need to store $\mathcal{O}(2R_\epsilon n^{1+\rho/2})$ complex numbers for the PLR compressed $\mathbf{G}$. Thus, the compression ratio is given by $\frac{2R_\epsilon n^{1+\rho/2}}{n^2} \sim n^{\rho/2-1}$.

Moreover, multiplying each block by a vector has asymptotic complexity $2R_\epsilon n^{1-\rho/2}$, so that the overall asymptotic complexity of the PLR matrix vector multiplication is given by $2R_\epsilon n^{1+\rho/2}$.

If $\rho = 1/2$ and $N \sim n^2$, we have that the asymptotic complexity of the PLR matrix-vector multiplication is given by $2R_\epsilon n^{5/4} \sim N^{5/8}$. If $\rho = 1$, the complexity becomes $\sim N^{3/4}$.

[10]The complexity overhead generated by smaller blocks close to the diagonal only increase the overall constant, not the asymptotic rate, much as in the fast multipole method.

Step	Analytic	Finite Differences
$\omega \sim \sqrt{n} \mid r \sim 1$	$\mathcal{O}(N^{5/8})$	$\mathcal{O}(N^{3/4})$
$\omega \sim n \mid r \sim 1$	$\mathcal{O}(N^{3/4})$	$\mathcal{O}(N^{7/8})$
$\omega \sim \sqrt{n} \mid r \sim \sqrt{n}$	$[\mathcal{O}(N^{5/8})] \to \mathcal{O}(N^{3/4})$	$[\mathcal{O}(N^{5/8})]$
$\omega \sim n \mid r \sim \sqrt{n}$	$[\mathcal{O}(N^{7/8})] \to \mathcal{O}(N)$	$[\mathcal{O}(N^{7/8})]$

Table 2.4: Compression scaling for the remote interactions, sampling a typical oscillatory kernel (Analytic) and using the finite differences approximation (FD). The observed pre-asymptotic complexities are in square brackets.

The estimate for the asymptotic complexity relies on a perfect knowledge of the phase functions, which is unrealistic for a finite difference approximation of the Green's functions. We empirically observe a deterioration of these scalings due to the discretization error, as shown in Table 2.4.

One possible practical fix for this problem is to allow the maximum ranks in the compression to grow as $\sqrt{n}$. This small adjustment allows us to reduce the complexity in our numerical experiments; though the theoretical predictions of those scalings (from a completely analogous analysis) are quite unfavorable. The scalings we observe numerically are reported in square brackets in the table. They are *pre-asymptotic* and misleadingly good: if the frequency and N were both increased, higher scaling exponents would be observed[11]. The correct numbers are without square brackets.

2.6 Computational Complexity

The complexities of the various steps of the algorithm were presented in section 2.1.3 and are summarized in Table 2.1. In this section we provide details about the heuristics and evidence supporting these complexity claims.

2.6.1 Computational cost

For the five-point discretization of the 2D Laplacian, the sparse LU factorization with nested dissection is known to have $\mathcal{O}(N^{3/2})$ complexity, and the back-substitution is known to have linear complexity up to logarithmic factors, see [74, 81]. Given that each sub-domain has size $\mathcal{O}(N/L)$, the factorization cost is $\mathcal{O}((N/L)^{3/2})$. Moreover, given that the factorizations are completely independent, they can be computed simultaneously in L different nodes. We used the stored LU factors to compute, by back-substitution, the local solutions needed for constructing the right-hand side of Eq. 2.41 and the reconstruction of the local solutions, leading to a complexity $\mathcal{O}(N\log(N)/L)$ per solve. The local solves are independent and can be performed in L different nodes.

[11]The same pre-asymptotic phenomenon occurs in the n-by-n FFT matrix: a block of size $n/2$ by $n/2$ will look like it has ϵ-rank $\mathcal{O}(n^{7/8})$, until $n \sim 2^{14}$ and beyond, where the ϵ-rank because very close to $n/4$.

To compute the Green's functions from the LU factors, we need to perform $\mathcal{O}(n)$ solves per layer. Each solve is completely independent of the others, and they can be carried in parallel in nL different processors. We point out that there is a clear trade-off between the parallel computation of the Green's functions and the communication cost. It is possible to compute all the solves of a layer in one node using shared memory parallelism, which reduces scalability; or compute the solves in several nodes, which increases scalability but increases the communication costs. The best balance between both approaches will heavily depend on the architecture and we leave the details out of this presentation.

We point out that the current complexity bottleneck is the extraction of the Green's functions. It may be possible to reduce the number of solves to a fractional power of n using randomized algorithms such as [91]); or to $\mathcal{O}(1)$ solves, if more information of the underlying PDE is available or easy to compute (see [10]).

Once the Green's matrices are computed, we use Alg. 5 to compress them. A simple complexity count indicates that the asymptotic complexity of the recursive adaptive compression is bounded by $\mathcal{O}((N\log(N)))$ for each integral kernel[12]. The compression of each integral kernel is independent of the others, and given that the Green's functions are local to each node, no communication is needed.

There are two limitations to the compression level of the blocks of $\underline{\underline{\mathbf{M}}}$. One is theoretical, and was explained in section 2.5.2; while the second is numerical. In general, we do not have access to the Green's functions, only to a numerical approximation, and the numerical errors present in those approximations hinder the compressibility of the operators for large ω. Faithfulness of the discretization is the reason why we considered milder scalings of ω as a function of n, such as $\omega \sim \sqrt{n}$, in the previous section [13].

With the scaling $\omega \sim \sqrt{n}$, we have that the complexity of matrix-vector product is dominated by $\mathcal{O}(N^{5/8})$ for each block of $\underline{\underline{\mathbf{M}}}$ (see subsection 2.5.2). Then, the overall complexity of the GMRES iteration is $\mathcal{O}(LN^{5/8})$, provided that the number of iterations remains bounded.

Within an HPC environment we are allowed to scale L, the number of sub-domains, as a small fractional power of N. If $L \sim N^{\delta}$, then the overall execution time of the online computation, take away the communication cost, is $\mathcal{O}(N^{\max(\delta+5/8,(1-\delta))})$. Hence, if $0 < \delta < 3/8$, then the online algorithm runs in sub-linear time. In particular, if $\delta = 3/16$, then the runtime of the online part of the solver becomes $\mathcal{O}(N^{13/16})$ plus a sub-linear communication cost.

If the matrix-vector product is $\mathcal{O}(N^{3/4})$ (a more representative figure in Table 2.4), then the same argument leads to $\delta = 1/8$ and an overall online runtime of $\mathcal{O}(N^{7/8})$.

[12] Assuming the svds operation is performed with a randomized SVD.

[13] To deduce the correct scaling between ω and n, we use the fact that the second order five point stencil scheme has a truncation error dominated by $h^2(\partial_x^4 + \partial_y^4)u \sim (\omega/c)^4 h^2$. Given that $h \sim 1/n$, we need $\omega \sim \sqrt{n}$ in order to have a bounded error in the approximation of the Green's function. In general, the scaling needed for a p-th order scheme to obtain a bounded truncation error is $\omega \sim n^{\frac{p}{p+2}}$.

2.6.2 Communication cost

We use a commonly-used communication cost model ([6, 53, 111, 136]) to perform the cost analysis. The model assumes that each process is only able to send or receive a single message at a time. When the messages has size m, the time to communicate that message is $\alpha + \beta m$. The parameter α is called latency, and represents the minimum time required to send an arbitrary message from one process to another, and is a constant overhead for any communication. The parameter β is called inverse bandwidth and represents the time needed to send one unit of data.

We assume that an interface restriction of the Green's functions can be stored in one node. Then, all the operations would need to be performed on distributed arrays, and communication between nodes would be needed at each GMRES iteration. However, in 2D, the data to be transmitted are only traces, so the overhead of transmitting such small amount of information several times would overshadow the complexity count. We gather the compressed Green's functions in a master node and use them as blocks to form the system $\underline{\underline{\mathbf{M}}}$. Moreover, we reuse the compressed blocks to form the matrices used in the preconditioner ($\underline{\mathbf{D}}^{-1}$ and $\underline{\mathbf{R}}$).

We suppose that the squared slowness model m and the sources $\mathbf{f}$ are already on the nodes, and that the local solutions for each source will remain on the nodes to be gathered in a post-processing step. In the context of seismic inversion this assumption is very natural, given that the model updates are performed locally as well.

Within the offline step, we suppose that the model is already known to the nodes (otherwise we would need to communicate the model to the nodes, incurring a cost $\mathcal{O}(\alpha + \beta N/L)$ for each layer). The factorization of the local problem, the extraction of the Green's functions and their compression are zero-communication processes. The compressed Green's functions are transmitted to the master node, with a maximum cost of $\mathcal{O}(\alpha + 4\beta N)$ for each layer. This cost is in general lower because of the compression of Green's function. In summary, the whole communication cost for the offline computations is dominated by $\mathcal{O}(L(\alpha + 4\beta N))$. Using the compression scaling for the Green's matrices in section 2.5.2 the communication cost can be reduced. However, it remains bounded from below by $\mathcal{O}(L\alpha + \beta N)$.

For the online computation, we suppose that the sources are already distributed (otherwise we would incur a cost $\mathcal{O}(\alpha + \beta N/L)$ for each layer). Once the local solves, which are zero-communication processes, are performed, the traces are sent to the master node, which implies a cost of $\mathcal{O}(L(\alpha + \beta N^{1/2}))$. The inversion of the discrete integral system is performed and the traces of the solution are sent back to the nodes, incurring another $\mathcal{O}(L(\alpha + \beta N^{1/2}))$ cost. Finally, from the traces, the local solutions are reconstructed and saved in memory for the final assembly in a post-processing step not accounted for here (otherwise, if the local solutions are sent back to the master node we would incur a $\mathcal{O}(\alpha + \beta N/L)$ cost per layer). In summary, the communication cost for the online computation is $\mathcal{O}(\alpha L + \beta L N^{1/2})$.

Finally, we point out that for the range of 2D numerical experiments performed in this Chapter, the communication cost was negligible with respect to the floating points operations.

2.7 Numerical Experiments

In this section we show numerical experiments that illustrate the behavior of the proposed algorithm. Any high frequency Helmholtz solver based on domain decomposition should ideally have three properties:

- the number of iterations should be independent of the number of the subdomains,
- the number of iterations should be independent of the frequency,
- the number of iterations should depend weakly on the roughness of the underlying model.

We show that our proposed algorithm satisfies the first two properties. The third property is also verified in cases of interest to geophysicists, though our solver is not expected to scale optimally (or even be accurate) in the case of resonant cavities.

We also show some empirical scalings supporting the sub-linear complexity claims for the GMRES iteration, and the sub-linear run time of the online part of the solver.

The code for the experiments was written in Python. We used the Anaconda implementation of Python 2.7.8 and the Numpy 1.8.2 and Scipy 0.14.0 libraries linked to OpenBLAS. All the experiments were carried out in 4 quad socket servers with AMD Opteron 6276 at 2.3 GHz and 256 Gbytes of RAM, linked with a gigabit Ethernet connection. All the system were preconditioned by two iterations of the Gauss-Seidel preconditioner.

2.7.1 Precomputation

To extract the Green's functions and to compute the local solutions, a pivoted sparse LU factorization is performed at each slab using UMFPACK [49], and the LU factors are stored in memory. The LU factors for each slab are independent of the others, so they can be stored in different cluster nodes. The local solves are local to each node, enhancing the granularity of the algorithm. Then the Green's function used to form the discrete integral are computed by solving the local problem with columns of the identity in the right-hand side. The computation of each column of the Green's function is independent of the rest and can be done in parallel (shared or distributed memory).

Once the Green's functions are computed, they are compressed in sparse PLR form, following Alg. 5. $\underline{\underline{\mathbf{M}}}$ and $\underline{\mathbf{R}}$ are implemented as a block matrices, in which each block is a PLR matrix in sparse form. The compression is accelerated using the randomized SVD [97]. The inversion of $\underline{\mathbf{D}}$ is implemented as a block back-substitution with compressed blocks.

We used a simple GMRES algorithm[14] (Algorithm 6.9 in [117]). Given that the number of iteration remains, in practice, bounded by ten, neither low rank update to the Hessenberg matrix nor re-start are necessary.

[14]Some authors refer to GMRES algorithm as the Generalized conjugate residual algorithm [25], [116]

N	$\omega/2\pi$ [Hz]	$L=8$	$L=16$	$L=32$	$L=64$	$L=128$
195×870	5.57	**(3)** 0.094	**(3)** 0.21	**(3)** 0.49	**(4)** 1.02	**(4)** 2.11
396×1720	7.71	**(3)** 0.14	**(3)** 0.32	**(3)** 0.69	**(4)** 1.55	**(4)** 3.47
792×3420	11.14	**(3)** 0.41	**(3)** 0.83	**(3)** 1.81	**(4)** 3.96	**(4)** 9.10
1586×6986	15.86	**(3)** 0.72	**(3)** 1.56	**(3)** 3.19	**(4)** 6.99	-

Table 2.5: Number of GMRES iterations (bold) required to reduce the relative residual to 10^{-7}, along with average execution time (in seconds) of one GMRES iteration for different N and L. The solver is applied to the smooth Marmousi2 model. The frequency is scaled such that $\omega \sim \sqrt{n}$. The matrices are compressed using $\epsilon = 10^{-9}/L$ and $\text{rank}_{\max} \sim \sqrt{n}$.

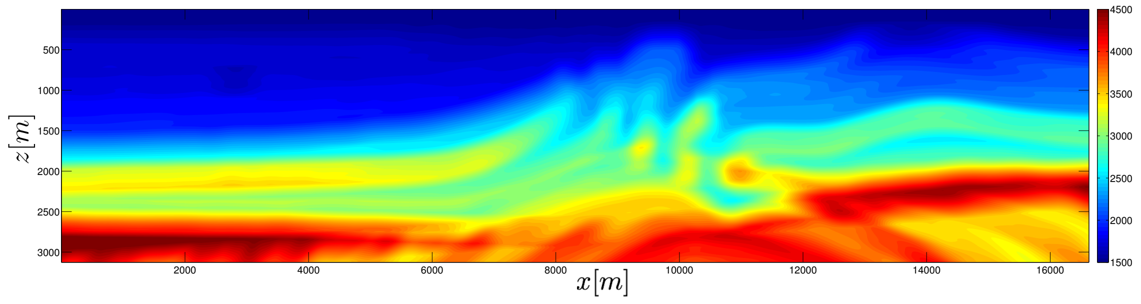


Figure 2-11: Smooth version of the Marmousi2 model. The model was smoothed with a box filter of size 375 meters.

2.7.2 Smooth Velocity Model

For the smooth velocity model, we choose a smoothed Marmousi2 model (see Fig. 2-11) that was partitioned in the longitudinal direction. Table 2.5 and Table 2.6 were generated by timing 200 randomly generated right-hand-sides. Table 2.5 shows the average runtime of one GMRES iteration, and Table 2.6 shows the average runtime of the online part of the solver. We can observe that for the smooth case the number of iterations is almost independent of the frequency and the number of sub-domains. In addition, the runtimes scales sub-linearly with respect to the number of volume unknowns.

2.7.3 Rough Velocity Model

In general, iterative solvers are highly sensitive to the roughness of the velocity model. Sharp transitions generates strong reflections that hinder the efficiency of iterative methods, increasing the number of iterations. Moreover, for large ω, the interaction of high frequency waves with short wavelength structures such as discontinuities, increases the reflections, further deteriorating the convergence rate.

The performance of the method proposed in this Chapter deteriorates only marginally

N	$\omega/2\pi$ [Hz]	$L = 8$	$L = 16$	$L = 32$	$L = 64$	$L = 128$
195×870	5.57	0.36	0.72	1.53	3.15	7.05
396×1720	7.71	0.79	1.26	2.35	4.99	11.03
792×3420	11.14	2.85	3.69	6.41	13.11	28.87
1586×6986	15.86	9.62	9.76	13.85	25.61	-

Table 2.6: Average execution time (in seconds) for the online computation, with a GMRES tolerance of 10^{-7}, for different N and L. The solver is applied to the smooth Marmousi2 model. The frequency is scaled such that $\omega \sim \sqrt{n}$. The matrices are compressed using $\epsilon = 10^{-9}/L$ and $\text{rank}_{\max} \sim \sqrt{n}$.

as a function of the frequency and number of subdomains.

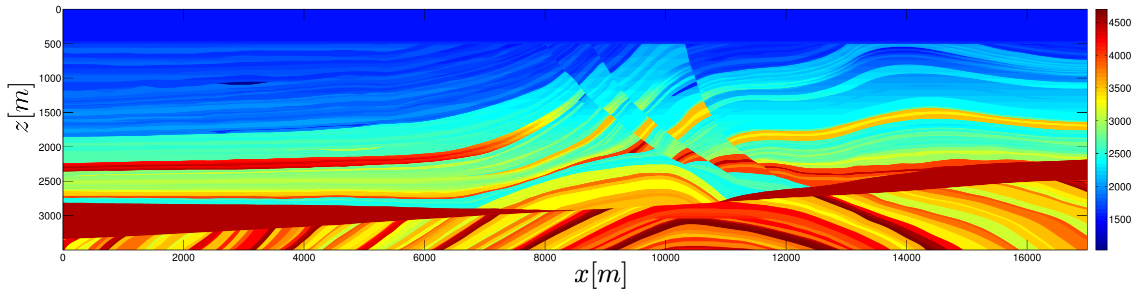


Figure 2-12: Geophysical benchmark model Marmousi2; the wave speed is in meters per second.

We use the Marmousi2 model [95], and another geophysical community benchmark, the BP 2004 model [19], depicted in Fig. 2-12 and Fig. 2-13 respectively, both are partitioned in the longitudinal direction in order to obtain a smaller integral system to solve.

Tables 2.7, 2.8, 2.9, and 2.10 were generated by running 200 randomly generated right hand sides inside the domain. The number of points for the perfectly matched layers is increased linearly with n. From Table 2.5 and Table 2.9 we can observe that, in general, the number of iteration to convergence grows slowly. We obtain slightly worse convergence rates if the number of points of the PML increases slowly or if it remains constant. This observation was already made in [128] and [112]. However, the asymptotic complexity behavior remains identical; only the constant changes.

The runtime for one GMRES iteration exhibits a slightly super-linear growth when L, the number of sub-domains, is increased to be a large fraction of n. This is explained by the different compression regimes for the interface operators and by the different tolerances for the compression of the blocks. When the interfaces are very close to each other the interactions between the source depths and the target depths increases, producing higher ranks on the diagonal blocks of the remote interactions.

Tables 2.8 and 2.10 show that the average runtime of each solve scales sublinearly with respect to the volume unknowns.

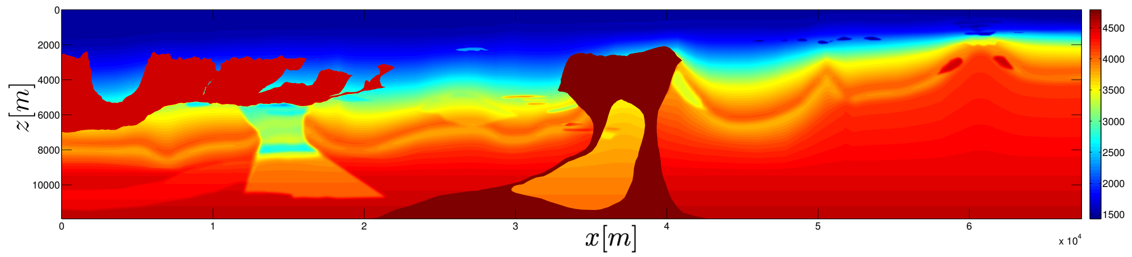


Figure 2-13: Geophysical benchmark model BP 2004 [19].

N	$\omega/2\pi$ [Hz]	$L = 8$	$L = 16$	$L = 32$	$L = 64$	$L = 128$
195×870	5.57	**(5)** 0.08	**(5)** 0.18	**(5)** 0.41	**(6)** 0.87	**(6)** 1.83
396×1720	7.71	**(5)** 0.17	**(6)** 0.41	**(6)** 0.88	**(6)** 2.06	**(7)** 4.57
792×3420	11.14	**(5)** 0.39	**(6)** 0.85	**(6)** 1.82	**(6)** 4.06	**(7)** 9.51
1586×6986	15.86	**(5)** 0.94	**(6)** 1.89	**(6)** 4.20	**(6)** 9.41	**(7)** 21.10

Table 2.7: Number of GMRES iterations (bold) required to reduce the relative residual to 10^{-7}, along with average execution time (in seconds) of one GMRES iteration for different N and L. The solver is applied to the Marmousi2 model. The frequency is scaled such that $\omega \sim \sqrt{n}$. The matrices are compressed using $\epsilon = 10^{-9}/L$ and $\text{rank}_{\max} \sim \sqrt{n}$.

Table 2.12 and Table 2.11 illustrate the maximum and minimum number of iterations and the average runtime of one GMRES iteration in the high frequency regime, i.e. $\omega \sim n$. We can observe that the number of iterations are slightly higher but with a slow growth. Moreover, we can observe the slightly sub-linear behavior of the runtimes, which we called the pre-asymptotic regime in Section 2.5.

Finally, Fig. 2-15 summarizes the scalings for the different stages of the computation for fixed L. For a small number of unknowns the cost is dominated by the solve of the integral system – a case in which the online part seems to have a sub-linear complexity. However, after this pre-asymptotic regime, the LU solve dominates the complexity, so it is lower-bounded by the $\mathcal{O}(N\log(N)/L)$ runtime. The GMRES timing is still in the pre-asymptotic regime, but the complexity would probably deteriorate to $\mathcal{O}(N^{3/4})$ for frequencies high enough. The complexity of the online part is non-optimal given that we only used a fixed number or layers and processors. (We do not advocate choosing such small L.)

N	$\omega/2\pi$ [Hz]	$L=8$	$L=16$	$L=32$	$L=64$	$L=128$
195×870	5.57	0.47	0.99	2.15	4.53	10.94
396×1720	7.71	1.30	2.36	5.04	12.66	29.26
792×3420	11.14	3.82	5.62	11.45	25.78	64.01
1586×6986	15.86	13.68	16.08	28.78	62.45	145.98

Table 2.8: Average execution time (in seconds) for the online computation, with a tolerance on the GMRES of 10^{-7}, for different N and L. The solver is applied to the Marmousi2 model. The frequency is scaled such that $\omega \sim \sqrt{n}$. The matrices are compressed using $\epsilon = 10^{-9}/L$ and $\text{rank}_{\max} \sim \sqrt{n}$.

N	$\omega/2\pi$	$L=8$	$L=16$	$L=32$	$L=64$	$L=128$
136×354	1.18	**(6)** 0.07	**(6)** 0.18	**(7)** 0.38	**(7)** 0.80	**(8)** 1.58
269×705	1.78	**(6)** 0.10	**(6)** 0.22	**(7)** 0.50	**(8)** 1.07	**(8)** 2.22
540×1411	2.50	**(6)** 0.22	**(7)** 0.52	**(7)** 1.22	**(7)** 2.80	**(9)** 5.97
1081×2823	3.56	**(7)** 0.38	**(7)** 0.87	**(8)** 1.93	**(8)** 4.33	**(9)** 10.07

Table 2.9: Number of GMRES iterations (bold) required to reduce the relative residual to 10^{-7}, along with average execution time (in seconds) of one GMRES iteration for different N and L. The solver is applied to the BP 2004 model. The frequency is scaled such that $\omega \sim \sqrt{n}$. The matrices are compressed using $\epsilon = 10^{-9}/L$ and $\text{rank}_{\max} \sim \sqrt{n}$.

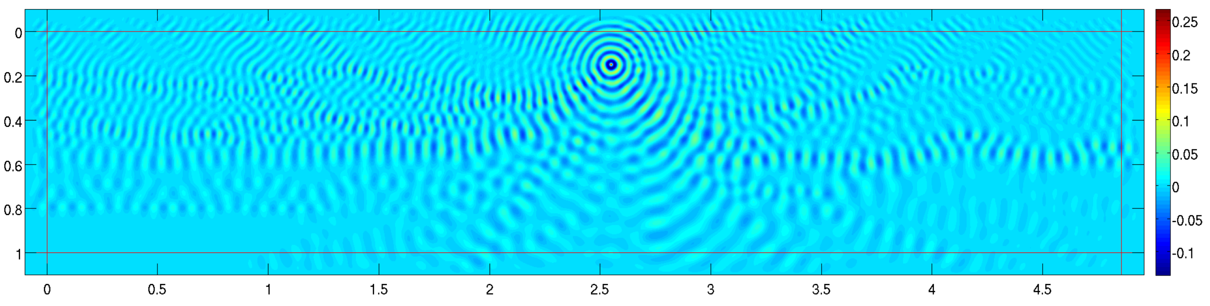


Figure 2-14: Real part of wavefield generated by a point source at 15.86 [Hz] with the Marmousi2 model [95] as a background model.

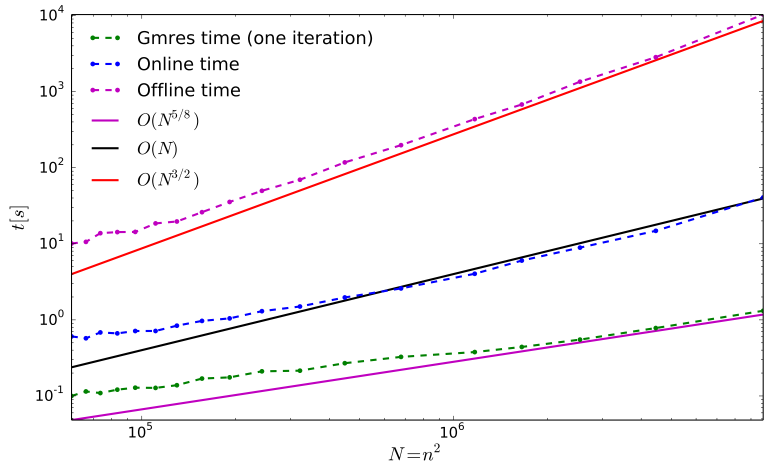


Figure 2-15: Run-time with their empirical complexities, for the Marmousi2 model with $L = 3$ and $\omega \sim \sqrt{n}$ and $\max_{\text{rank}} \sim \sqrt{n}$.

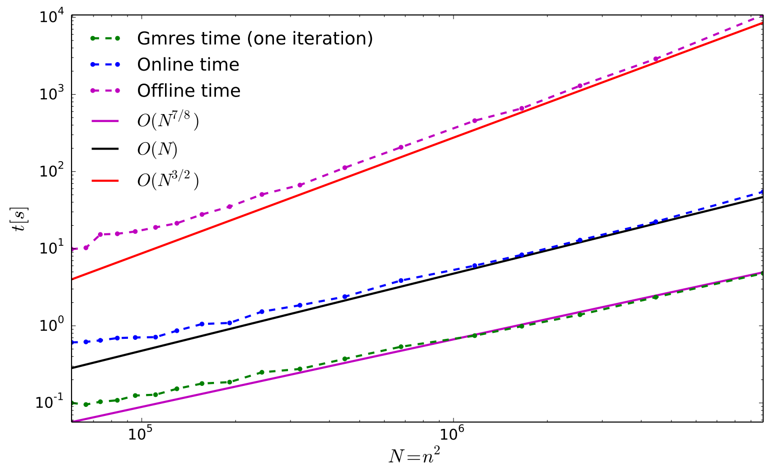


Figure 2-16: Run-times and empirical complexities, for the Marmousi2 model with $L = 3$ and $\omega \sim n$ and $\max_{\text{rank}} \sim \sqrt{n}$.

N	$\omega/2\pi$ [Hz]	$L = 8$	$L = 16$	$L = 32$	$L = 64$	$L = 128$
136×354	1.18	0.45	1.09	2.69	5.67	12.44
269×705	1.78	0.70	1.44	3.45	7.86	17.72
540×1411	2.50	1.85	3.74	8.83	20.09	48.86
1081×2823	3.56	4.93	7.87	15.91	35.69	83.63

Table 2.10: Average execution time (in seconds) for the online computation, with a tolerance on the GMRES of 10^{-7}, for different N and L. The solver is applied to the BP 2004 model. The frequency is scaled such that $\omega \sim \sqrt{n}$. The matrices are compressed using $\epsilon = 10^{-9}/L$ and $\text{rank}_{\max} \sim \sqrt{n}$.

N	$\omega/2\pi$ [Hz]	$L = 8$	$L = 16$	$L = 32$	$L = 64$	$L = 128$
195×870	7.2	**(5)** 0.08	**(5)** 0.19	**(6)** 0.40	**(6)** 0.83	**(6)** 1.74
396×1720	15	**(5)** 0.23	**(6)** 0.49	**(6** 0.98	**(6)** 2.17	**(7)** 5.13
792×3420	30	**(6)** 0.69	**(6)** 1.54	**(7)** 2.48	**(7)** 4.93	**(8)** 10.64
1586×6986	60	**(6)** 2.29	**(6)** 5.04	**(7)** 8.47	**(8)** 14.94	-

Table 2.11: Number of GMRES iterations (bold) required to reduce the relative residual to 10^{-7}, along with average execution time (in seconds) of one GMRES iteration for different N and L. The solver is applied to the Marmousi2 model. The frequency is scaled such that $\omega \sim n$. The matrices are compressed using $\epsilon = 10^{-9}/L$ and $\text{rank}_{\max} \sim \sqrt{n}$.

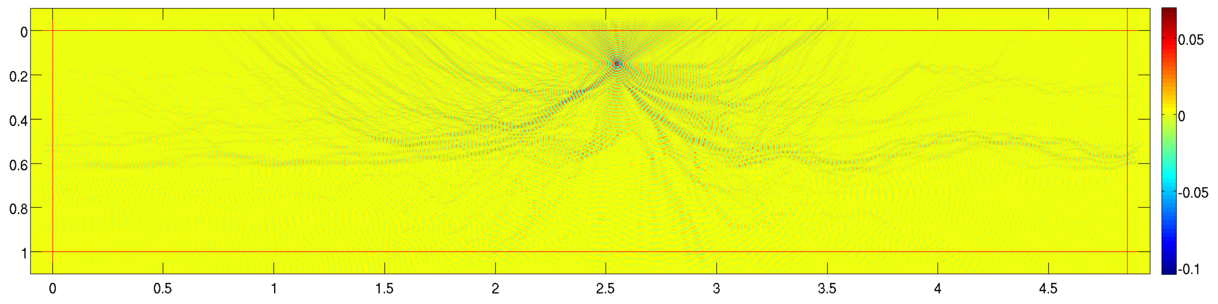


Figure 2-17: Wavefield generated by a point source at 60 [Hz] with the Marmousi2 model [95] as a background model; the red boundary indicates the global PML used.

N	$\omega/2\pi$ [Hz]	$L = 8$	$L = 16$	$L = 32$	$L = 64$	$L = 128$
136×354	1.4	**(6)** 0.073	**(7)** 0.18	**(8)** 0.37	**(8)** 0.84	**(9)** 1.52
269×705	2.7	**(7)** 0.10	**(7)** 0.23	**(7)** 0.50	**(9)** 1.10	**(9)** 2.14
540×1411	5.5	**(7)** 0.32	**(8)** 0.64	**(9)** 1.30	**(9)** 3.06	**(12)** 6.22
1081×2823	11.2	**(7)** 0.87	**(8)** 1.46	**(9)** 2.78	**(10)** 5.75	**(12)** 12.7

Table 2.12: Number of GMRES iterations (bold) required to reduce the relative residual to 10^{-7}, along with average execution time (in seconds) of one GMRES iteration for different N and L. The solver is applied to the BP 2004 model. The frequency is scaled such that $\omega \sim n$. The matrices are compressed using $\epsilon = 10^{-9}/L$ and $\text{rank}_{\text{max}} \sim \sqrt{n}$.

Appendix

2.A Discretization

Computations for Eq. 2.26 in 1D

The stencils for the derivatives of G and u are given by the discrete Green's representation formula. We present an example in 1D, which is easily generalized to higher dimension. Let $u = \{u_i\}_{i=0}^{n+1}$ and $v = \{v_i\}_{i=0}^{n+1}$. Define $\Omega = \{1, ...n\}$ with the discrete inner product

$$\langle u, v\rangle_\Omega = \sum_{i=1}^{n} u_i v_i.$$

Let $\triangle^h u$ be the three-point stencil approximation of the second derivative. I.e.

$$\left(\triangle^h u\right)_i = u_{i-1} - 2u_i + u_{i+1}.$$

We can use summation by parts to obtain the expression

$$\langle \triangle^h u, v\rangle_\Omega = \langle u, \triangle^h v\rangle_\Omega - v_0(u_1 - u_0) + u_0(v_1 - v_0) + v_{n+1}(u_{n+1} - u_n) - u_{n+1}(v_{n+1} - v_n). \tag{2.99}$$

The formula above provides the differentiation rule to compute the traces and the derivatives of u and G at the interfaces. For example, let G_i^k be the solution to 1D discrete Helmholtz operator defined by $HG^k = (-\triangle^h - m\omega^2)G^k = \delta_i^k$, for $j \in [1, ..., n]$. Then

$$u_k = \langle HG^k, u\rangle_\Omega = \langle G^k, Hu\rangle_\Omega - G_0^k(u_1 - u_0) + u_0(G_1^k - G_0^k) + G_{n+1}^k(u_{n+1} - u_n) - u_{n+1}(G_{n+1}^k - G_n^k). \tag{2.100}$$

To simplify the notations we define

$$\partial^+ G_0 = G_1 - G_0, \qquad \partial^- G_{n+1} = G_{n+1} - G_n, \tag{2.101}$$

which are upwind derivatives with respect to Ω, i.e. they look for information inside Ω. Let us consider a partition of $\Omega = \Omega^1 \cup \Omega^2$, where $\Omega^1 = \{1, n^1\}$ and $\Omega^2 = \{n^1 + 1, n^2\}$. We can define local inner products between u and v analogously,

$$\langle u, v\rangle_{\Omega^1} = \sum_{i=1}^{n^1} u_i v_i, \qquad \langle u, v\rangle_{\Omega^2} = \sum_{i=n^1+1}^{n^2} u_i v_i,$$

in such a way that

$$\langle u, v\rangle_{\Omega^1} + \langle u, v\rangle_{\Omega^2} = \langle u, v\rangle_{\Omega}.$$

We can use summation by parts in each sub-domain, obtaining

$$\langle HG, u\rangle_{\Omega^1} = \langle G, Hu\rangle_{\Omega^1} - G_0\partial^+ u_0 + u_0\partial^+ G_0 + G_{n^1+1}\partial^- u_{n^1+1} - u_{n^1+1}\partial^- G_{n^1+1}, \tag{2.102}$$

$$\langle HG, u\rangle_{\Omega^2} = \langle G, Hu\rangle_{\Omega^2} - G_{n^1}\partial^+ u_{n^1} + u_{n^1}\partial^+ G_{n^1} + G_{n^2+1}\partial^- u_{n^2+1} - u_{n^2+1}\partial^- G_{n^2+1}, \tag{2.103}$$

which are the 1D version of Eq. 2.26.

Computations for Eq. 2.26 in 2D

The 2D case is more complex given that we have to consider the PML in the x direction when integrating by parts. To simplify the proofs, we need to introduce the symmetric formulation of the PML's for the Helmholtz equation. We show in this section that the symmetric and unsymmetric formulations lead to the same Green's representation formula. In addition, we present Eq. 2.106, which links the Green's functions of both formulations. In Appendix 2.C we use the symmetric formulation in all of the proofs; however, owing to Eq. 2.106 the proofs are valid for the unsymmetric formulation as well.

Following [58] we define the symmetrized Helmholtz equation with PML's given by

$$-\left(\partial_x \frac{\alpha_x(\mathbf{x})}{\alpha_z(\mathbf{x})}\partial_x + \partial_z \frac{\alpha_z(\mathbf{x})}{\alpha_x(\mathbf{x})}\partial_z\right) u - \frac{m\omega^2}{\alpha_x(\mathbf{x})\alpha_z(\mathbf{x})} u = \frac{f}{\alpha_x(\mathbf{x})\alpha_z(\mathbf{x})}, \tag{2.104}$$

which is obtained by dividing Eq. 2.10 by $\alpha_x(\mathbf{x})\alpha_z(\mathbf{x})$ and using the fact that α_x is independent of z, and α_z is independent of x. We can use the same discretization used in Eq. 2.13 to obtain the system

$$\begin{aligned}(\mathbf{H}^{\mathrm{s}}\mathbf{u})_{i,j} = &-\frac{1}{h^2}\left(\frac{\alpha_x(\mathbf{x}_{i+1/2,j})}{\alpha_z(\mathbf{x}_{i+1/2,j})}(\mathbf{u}_{i+1,j} - \mathbf{u}_{i,j}) - \frac{\alpha_x(\mathbf{x}_{i-1/2,j})}{\alpha_z(\mathbf{x}_{i-1/2,j})}(\mathbf{u}_{i,j} - \mathbf{u}_{i-1,j})\right) \\ &-\frac{1}{h^2}\left(\frac{\alpha_z(\mathbf{x}_{i,j+1/2})}{\alpha_x(\mathbf{x}_{i,j+1/2})}(\mathbf{u}_{i,j+1} - \mathbf{u}_{i,j}) - \frac{\alpha_z(\mathbf{x}_{i,j-1/2})}{\alpha_x(\mathbf{x}_{i,j-1/2})}(\mathbf{u}_{i,j} - \mathbf{u}_{i,j-1})\right) \\ &-\frac{m(\mathbf{x}_{i,j})\omega^2}{\alpha_x(\mathbf{x}_{i,j})\alpha_z(\mathbf{x}_{i,j})}\mathbf{u}_{i,j} = \mathbf{f}_{i,j},\end{aligned} \tag{2.105}$$

in which we used the fact that the support of $\mathbf{f}$ is included in the physical domain, where α_x and α_z are equal to 1. Moreover, from Eq. 2.104 and Eq. 2.16, it is easy to prove that the Green's function associated to $\mathbf{H}^{\mathrm{s}}$ satisfies

$$\mathbf{G}(\mathbf{x}, \mathbf{y})\alpha_x(\mathbf{y})\alpha_z(\mathbf{y}) = \mathbf{G}^{\mathrm{s}}(\mathbf{x}, \mathbf{y}). \tag{2.106}$$

Given that $\mathbf{f}$ is supported inside the physical domain we have that $\alpha_x(\mathbf{y})\alpha_z(\mathbf{y})\mathbf{f}(\mathbf{y}) = \mathbf{f}(\mathbf{y})$, then applying $\mathbf{G}$ and $\mathbf{G}^{\mathrm{s}}$ to $\mathbf{f}$ yield the same answer, i.e., $\mathbf{u}$ and $\mathbf{u}^s$ (the solution to the symmetrized system) are identical.

To deduce Eq. 2.26, we follow the same computation as in the 1D case. We use the discrete ℓ^2 inner product, and we integrate by parts to obtain

$$
\begin{aligned}
\langle \mathbf{H}^{\mathbf{s}}\mathbf{u}, \mathbf{v}\rangle &= \sum_{i,j=1}^{n_x,n_z} (\mathbf{H}^{\mathbf{s}}\mathbf{u})_{i,j}\,\mathbf{v}_{i,j}, \\
&= \langle \mathbf{u}, \mathbf{H}^{\mathbf{s}}\mathbf{v}\rangle \\
&\quad - \sum_{j=1}^{n_z}\left[\frac{\alpha_x(\mathbf{x}_{1/2,j})}{\alpha_z(\mathbf{x}_{1/2,j})}\left(\mathbf{u}_{0,j}\partial_x^+\mathbf{v}_{0,j} - \mathbf{v}_{0,j}\partial_x^+\mathbf{u}_{0,j}\right)\right. \\
&\quad \left. + \frac{\alpha_x(\mathbf{x}_{n_x+1/2,j})}{\alpha_z(\mathbf{x}_{n_x+1/2,j})}\left(\mathbf{u}_{n_x+1,j}\partial_x^-\mathbf{v}_{n_x+1,j} - \mathbf{v}_{n_x+1,j}\partial_x^-\mathbf{u}_{n_x+1,j}\right)\right] \\
&\quad - \sum_{i=1}^{n_x}\left[\frac{\alpha_z(\mathbf{x}_{i,1/2})}{\alpha_x(\mathbf{x}_{i,1/2})}\left(\mathbf{u}_{i,0}\partial_z^+\mathbf{v}_{i,0} - \mathbf{v}_{i,0}\partial_z^+\mathbf{u}_{i,0}\right)\right. \\
&\quad \left. + \frac{\alpha_z(\mathbf{x}_{i,n_z+1/2})}{\alpha_x(\mathbf{x}_{i,n_z+1/2})}\left(\mathbf{u}_{i,n_z+1}\partial_z^-\mathbf{v}_{i,n_z+1} - \mathbf{v}_{i,n_z+1}\partial_z^-\mathbf{u}_{i,n_z+1}\right)\right].
\end{aligned}
$$

This is the general formula for the GRF. Moreover, given the nature of the layered partition, we can further simplify this expression. In each subdomain we have

$$
\begin{aligned}
\langle \mathbf{H}^{\mathbf{s}}\mathbf{u}, \mathbf{v}\rangle_{\Omega^\ell} &= \sum_{i=-n_{\mathrm{pml}}+1}^{n_x+n_{\mathrm{pml}}}\sum_{j=1}^{n_z} (\mathbf{H}^{\mathbf{s}}\mathbf{u})_{i,j}\,\mathbf{v}_{i,j}, \\
&= \langle \mathbf{u}, \mathbf{H}^{\mathbf{s}}\mathbf{v}\rangle_{\Omega^\ell} \\
&\quad - \sum_{j=1}^{n_z}\left[\frac{\alpha_x(\mathbf{x}_{1/2-n_{\mathrm{pml}}),j})}{\alpha_z(\mathbf{x}_{1/2-n_{\mathrm{pml}}),j})}\left(\mathbf{u}_{-n_{\mathrm{pml}},j}\partial_x^+\mathbf{v}_{-n_{\mathrm{pml}},j} - \mathbf{v}_{-n_{\mathrm{pml}},j}\partial_x^+\mathbf{u}_{-n_{\mathrm{pml}},j}\right)\right. \\
&\quad \left. + \frac{\alpha_x(\mathbf{x}_{n_x+n_{\mathrm{pml}}+1/2,j})}{\alpha_z(\mathbf{x}_{n_x+n_{\mathrm{pml}}+1/2,j})}\left(\mathbf{u}_{n_x+n_{\mathrm{pml}}+1,j}\partial_x^-\mathbf{v}_{n_x+n_{\mathrm{pml}}+1,j} - \mathbf{v}_{n_x+n_{\mathrm{pml}}+1,j}\partial_x^-\mathbf{u}_{n_x+n_{\mathrm{pml}}+1,j}\right)\right] \\
&\quad - \sum_{i=-n_{\mathrm{npml}}+1}^{n_x+n_{\mathrm{npml}}}\left[\frac{\alpha_z(\mathbf{x}_{i,1/2})}{\alpha_x(\mathbf{x}_{i,1/2})}\left(\mathbf{u}_{i,0}\partial_z^+\mathbf{v}_{i,0} - \mathbf{v}_{i,0}\partial_z^+\mathbf{u}_{i,0}\right)\right. \\
&\quad \left. + \frac{\alpha_z(\mathbf{x}_{i,n_z+1/2})}{\alpha_x(\mathbf{x}_{i,n_z+1/2})}\left(\mathbf{u}_{i,n_z+1}\partial_z^-\mathbf{v}_{i,n_z+1} - \mathbf{v}_{i,n_z+1}\partial_z^-\mathbf{u}_{i,n_z+1}\right)\right].
\end{aligned}
$$

In each subdomain this expression is never evaluated inside the PML for z, hence $\alpha_z(\mathbf{x}) = 1$. Moreover, if $\mathbf{u}$ and $\mathbf{v}$ both satisfy homogeneous Dirichlet boundary conditions at the vertical boundaries ($i = -n_{\mathrm{pml}}$ and $i = n_x + n_{\mathrm{pml}} + 1$, which are imposed in the formulation of the PML), we obtain

$$\begin{aligned}\langle \mathbf{H^s u}, \mathbf{v}\rangle_{\Omega^\ell} = \langle \mathbf{u}, \mathbf{H^s v}\rangle_{\Omega^\ell} \\ &- \sum_{i=-n_{\mathrm{pml}}+1}^{n_x+n_{\mathrm{pml}}} \left[\frac{1}{\alpha_x(\mathbf{x}_{i,1/2})} \left(\mathbf{u}_{i,0} \partial_z^+ \mathbf{v}_{i,0} - \mathbf{v}_{i,0} \partial_z^+ \mathbf{u}_{i,0} \right) \right. \\ &\left. - \frac{1}{\alpha_x(\mathbf{x}_{i,n_z+1/2})} \left(\mathbf{u}_{i,n_z+1} \partial_z^- \mathbf{v}_{i,n_z+1} - \mathbf{v}_{i,n_z+1} \partial_z^- \mathbf{u}_{i,n_z+1} \right) \right].\end{aligned}$$

We can then replace $\mathbf{u}$ and $\mathbf{v}$ by $\mathbf{u}^s$, the solution to $\mathbf{H^s u}^s = \mathbf{f}$, and $\mathbf{G^s}$. By construction both satisfy homogeneous Dirichlet boundary conditions at $i = -n_{\mathrm{pml}}$ and $i = n_x + 1 + n_{\mathrm{pml}}$, therefore, we obtain

$$\begin{aligned}\langle \mathbf{H^s G^s}, \mathbf{u}^s\rangle_{\Omega^\ell} &= \mathbf{u}^s \\ &= \langle \mathbf{G^s}, \mathbf{f}\rangle_{\Omega^\ell} - \sum_{i=-n_{\mathrm{pml}}+1}^{n_x+n_{\mathrm{pml}}} \left[\frac{1}{\alpha_x(\mathbf{x}_{i,1/2})} \left(\mathbf{G^s}_{i,0} \partial_z^+ \mathbf{u}^s_{i,0} - \mathbf{u}^s_{i,0} \partial_z^+ \mathbf{G^s}_{i,0} \right) \right. \qquad (2.107) \\ &\left. - \frac{1}{\alpha_x(\mathbf{x}_{i,n_z+1/2})} \left(\mathbf{G^s}_{i,n_z+1} \partial_z^- \mathbf{u}^s_{i,n_z+1} - \mathbf{u}^s_{i,n_z+1} \partial_z^- \mathbf{G^s}_{i,n_z+1} \right) \right]. \qquad (2.108)\end{aligned}$$

Moreover, using the relation between both Green's functions, the independence of α_x with respect to z, the point-wise equality between $\mathbf{u}$ and $\mathbf{u}^s$, and the properties of the support of $\mathbf{f}$; we obtain the Green's representation formula for the layered partition as

$$\mathbf{u} = \langle \mathbf{G}, \mathbf{f}\rangle + \sum_{i=-n_{\mathrm{pml}}+1}^{n_x+n_{\mathrm{pml}}} \left[\left(-\mathbf{G}_{i,0} \partial_z^+ \mathbf{u}_{i,0} + \mathbf{u}_{i,0} \partial_z^+ \mathbf{G}_{i,0} \right) + \left(\mathbf{G}_{i,n_z+1} \partial_z^- \mathbf{u}_{i,n_z+1} - \mathbf{u}_{i,n_z+1} \partial_z^- \mathbf{G}_{i,n_z+1} \right) \right]. \qquad (2.109)$$

It is the discrete version of Eq. 2.6.

2.B Triangular and block triangular matrices

In this section we introduce the necessary tools for the proofs in Appendix 2.C. They are by no means original.

Let H be the symmetric discretized Helmholtz operator in 1D, where

$$H = \begin{bmatrix} b_1 & c_1 & 0 & \dots & 0 \\ a_1 & b_2 & c_2 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & a_{n-2} & b_{n-1} & c_{n-1} \\ 0 & \dots & 0 & a_{n-1} & b_n \end{bmatrix}, \qquad (2.110)$$

in which $a = c$ by symmetry. We denote by G the inverse of H.

We follow [141] in writing a simple recurrence formula based on the minors

$$\theta_{-1} = 0, \qquad \theta_0 = 1, \qquad \theta_i = b_i\theta_{i-1} - a_i c_{i-1}\theta_{i-2}, \tag{2.111}$$
$$\phi_{n+2} = 0, \qquad \phi_{n+1} = 1, \qquad \phi_i = b_i\phi_{i+1} - c_i a_{i+1}\phi_{i+2}. \tag{2.112}$$

Lemma 8 (Lemma 2 in [141]). *We have the following identity:*

$$\theta_i\phi_{i+1} - a_{i+1}c_i\theta_{i-1}\phi_{i+2} = \theta_n, \qquad \forall 1 \le i \le n. \tag{2.113}$$

Theorem 2 (Theorem 1 in [141]). *If H is nonsingular, then its inverse is given by*

$$(H^{-1})_{i,j} = \begin{cases} (-1)^{i+j}\left(\Pi_{k=i}^{\ell-1}c_k\right)\frac{\theta_{i-1}\phi_{j+1}}{\theta_n} & \text{if } i < j, \\ \frac{\theta_{i-1}\phi_{i+1}}{\theta_n} & \text{if } i = j, \\ (-1)^{i+j}\left(\Pi_{k=j+1}^{i}a_k\right)\frac{\theta_{j-1}\phi_{i+1}}{\theta_n} & \text{if } i > j. \end{cases} \tag{2.114}$$

Proposition 2. *(Rank-one property). We have*

$$\left(H^{-1}_{i+1,i+1}\right)^{-1} H^{-1}_{i+1,i}H^{-1}_{i+1,k} = H^{-1}_{i,k}, \qquad \text{for } i < k, \tag{2.115}$$
$$\left(H^{-1}_{i-1,i-1}\right)^{-1} H^{-1}_{i-1,i}H^{-1}_{i-1,k} = H^{-1}_{i,k}, \qquad \text{for } i > k. \tag{2.116}$$

Moreover

$$\left[\left(H^{-1}_{i+1,i+1}\right)^{-1} H^{-1}_{i+1,i}\right] H^{-1}_{i+1,i} = H^{-1}_{i,i} + c_i^{-1}\left[\left(H^{-1}_{i+1,i+1}\right)^{-1} H^{-1}_{i+1,i}\right], \tag{2.117}$$
$$\left[\left(H^{-1}_{i-1,i-1}\right)^{-1} H^{-1}_{i-1,i}\right] H^{-1}_{i-1,i} = H^{-1}_{i,i} + a_i^{-1}\left[\left(H^{-1}_{i-1,i-1}\right)^{-1} H^{-1}_{i-1,i}\right]. \tag{2.118}$$

Proof. Eq. 2.115 and Eq. 2.116 are direct application of the expression for the inverse of H given by Thm. 2.

We only prove Eq. 2.117 – the proof of Eq. 2.118 is analogous. Using the expression of the inverse given by Thm. 2 we have

$$\left(H^{-1}_{i+1,i+1}\right)^{-1} H^{-1}_{i+1,i} = -c_i\frac{\theta_{i-1}}{\theta_i}. \tag{2.119}$$

Thus,

$$\begin{aligned} \left[\left(H^{-1}_{i+1,i+1}\right)^{-1} H^{-1}_{i+1,i}\right] H^{-1}_{i+1,i} &= c_i\frac{\theta_{i-1}}{\theta_i}a_{i+1}\frac{\theta_{i-1}\phi_{i+2}}{\theta_n} \\ &= \frac{\theta_{i-1}\phi_{i+1}}{\theta_n} - \frac{\theta_{i-1}}{\theta_i}\left(\frac{\theta_i\phi_{i+1} - a_{i+1}c_i\theta_{i-1}\phi_{i+1}}{\theta_n}\right) \\ &= \frac{\theta_{i-1}\phi_{i+1}}{\theta_n} - \frac{\theta_{i-1}}{\theta_i} \\ &= H^{-1}_{i,i} + c_i^{-1}\left[\left(H^{-1}_{i+1,i+1}\right)^{-1} H^{-1}_{i+1,i}\right]. \end{aligned}$$

Above, we used Lemma 8 and the expression for the inverse given by Thm. 2. $\square$

For the 2D case, we introduce the same ordering as in [58], where we increase the index in x first,

$$\mathbf{u} = (u_{1,1}, u_{2,1}, ..., u_{n_x,1}, u_{1,2}, ..., u_{n_x,n_z}). \tag{2.120}$$

For simplicity of exposition, and only for this section, we do not take into account the degrees of freedom within the PML. Let $\mathbf{H}$ be the discrete symmetric Helmholtz operator in 2D (Eq. 2.104), which we rewrite as

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_1 & \mathbf{C}_1 & 0 & \dots & 0 \\ \mathbf{C}_1 & \mathbf{H}_2 & \mathbf{C}_2 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & \mathbf{C}_{n_z-2} & \mathbf{H}_{n_z-1} & \mathbf{C}_{n_z-1} \\ 0 & \dots & 0 & \mathbf{C}_{n_z-1} & \mathbf{H}_{n_z} \end{bmatrix}, \tag{2.121}$$

in which each sub-block is a matrix in $\mathbb{C}^{n_x \times n_x}$ matrix (or a matrix in $\mathbb{C}^{n_x+2n_{\text{pml}},n_x+2n_{\text{pml}}}$ if we count the degrees of freedom within the PML). Each $\mathbf{C}_i$ is a constant times the identity, and each $\mathbf{H}_i$ is a tridiagonal symmetric matrix. The ordering of the unknowns implies that every block in $\mathbf{H}$ correspond to degrees of freedom with fixed depth (fixed z).

In order to prove the equivalent of the rank-one property in the 2D case we follow [17] and [99].

Definition 10. *Let Δ_i and Σ_i be defined by the following recurrence relations*

$$\Delta_1 = \mathbf{H}_1, \qquad \Delta_i = \mathbf{H}_i - \mathbf{C}_i \left(\Delta_{i-1}\right)^{-1} \mathbf{C}_i^t; \tag{2.122}$$

$$\Sigma_{n_z} = \mathbf{H}_{n_z}, \qquad \Sigma_i = \mathbf{H}_i - \mathbf{C}_{i+1}^t \left(\Sigma_{i+1}\right)^{-1} \mathbf{C}_{i+1}. \tag{2.123}$$

Proposition 3. *$\mathbf{H}$ is proper*[15]*, and its inverse is given by*

$$\mathbf{H}^{-1}_{j,k} = \begin{cases} \mathbf{U}_j \mathbf{V}_k^t & \text{if } j \le k, \\ \mathbf{V}_j \mathbf{U}_k^t & \text{if } j \ge k. \end{cases} \tag{2.124}$$

where $\mathbf{U}_j = \mathbf{C}_j^{-t}\Delta_{j-1}...\mathbf{C}_2^{-t}\Delta_1$ and $\mathbf{V}_j^t = \Sigma_1^{-1}\mathbf{C}_2^t...\mathbf{C}_j\Sigma_j^{-1}$.

Proof. $\mathbf{H}$ is proper because $\mathbf{C}_j$ are invertible; Eq. 2.124 is a direct application of Theorem 3.4 in [99]. $\square$

Proposition 4. $\left(\mathbf{H}^{-1}_{j+1,j+1}\right)^{-1}\mathbf{H}^{-1}_{j+1,j}$ *is symmetric and we have*

$$\left(\mathbf{H}^{-1}_{j+1,j+1}\right)^{-1}\mathbf{H}^{-1}_{j+1,j}\mathbf{H}^{-1}_{j+1,k} = \mathbf{H}^{-1}_{j,k}, \qquad \text{for } j < k. \tag{2.125}$$

Moreover,

$$\left(\mathbf{H}^{-1}_{j-1,j-1}\right)^{-1}\mathbf{H}^{-1}_{j-1,j}\mathbf{H}^{-1}_{j-1,k} = \mathbf{H}^{-1}_{j,k}, \qquad \text{for } j > k. \tag{2.126}$$

[15] A block Hessenberg matrix, with invertible upper and lower diagonal blocks. See Def. 2.2 in [17].

Proof. First, it is easy to prove that Δ_j are symmetric matrices using an inductive argument and Def. 10. Then, using Eq. 2.124 we have

$$\begin{aligned}
\left(\mathbf{H}^{-1}_{j+1,j+1}\right)^{-1}\mathbf{H}^{-1}_{j+1,j} &= \left(\mathbf{V}_{j+1}\mathbf{U}^t_{j+1}\right)^{-1}\mathbf{V}_{j+1}\mathbf{U}^t_j,\\
&= \mathbf{U}^{-t}_{j+1}\mathbf{U}^t_j,\\
&= \left(\mathbf{C}^{-t}_{j+1}\Delta_j...\mathbf{C}^{-t}_2\Delta_1\right)^{-t}\left(\mathbf{C}^{-t}_j\Delta_{j-1}...\mathbf{C}^{-t}_2\Delta_1\right)^t,\\
&= \mathbf{C}_{j+1}\Delta^{-t}_j,\\
&= \mathbf{C}_{j+1}\Delta^{-1}_j. \qquad (2.127)
\end{aligned}$$

Using the symmetry of Δ_j and the fact that $\mathbf{C}_{j+1}$ is an identity times a constant, we obtain the desired symmetry property.

Finally, a simple computation using Proposition 3 leads to Eq. 2.125. The proof of Eq. 2.126 is analogous. □

Definition 11. *Let $\mathbf{D}_j$ and $\mathbf{E}_j$ be defined by the following recurrences*

$$\mathbf{D}_1 = \mathbf{I}, \qquad \mathbf{D}_2 = -\mathbf{C}^{-1}_1\mathbf{H}_1, \qquad \mathbf{D}_j = -\left(\mathbf{D}_{j-2}\mathbf{C}_{j-1} + \mathbf{D}_{j-1}\mathbf{H}_{j-1}\right)\mathbf{C}^{-1}_j \quad j = 3,...,n_z; \qquad (2.128)$$

$$\mathbf{E}_{n_z} = \mathbf{j}, \quad \mathbf{E}_{n_z-1} = -\mathbf{H}_{n_z}\mathbf{C}^{-1}_{n_z-1}, \quad \mathbf{E}_j = -\mathbf{C}^{-1}_j\left(\mathbf{H}_{j+1}\mathbf{E}_{j+1} + \mathbf{C}_{j+1}\mathbf{E}_{j+2}\right) \quad j = n_z-2,...,1, \qquad (2.129)$$

and define the generalized Casorati determinant by

$$\mathbf{R}_j = \mathbf{C}_{j-1}\left(\mathbf{D}_j\mathbf{E}_{j-1} - \mathbf{D}_{j-1}\mathbf{E}_j\right), \qquad j = 2,...,n_z; \qquad \mathbf{R} := \mathbf{R}_{n_z}. \qquad (2.130)$$

Remark 3. *We note that $\mathbf{D}_j$ and $\mathbf{E}_j$ are invertible. Indeed, we can see that $\mathbf{D}_j$ has n linearly independent solutions to the three-term recurrence. Then the determinant of $\mathbf{D}_j$ is always different from zero. The same is true for $\mathbf{E}_j$ but using the backwards recurrence.*

Proposition 5. *For the sequence of matrices $\mathbf{D}_j$ and $\mathbf{E}_j$, its generalized Casorati determinant is constant.*

Proof. We compute

$$\begin{aligned}
\mathbf{R}_j - \mathbf{R}_{j+1} &= \mathbf{C}_{j-1}\left(\mathbf{D}_j\mathbf{E}_{j-1} - \mathbf{D}_{j-1}\mathbf{E}_j\right) - \mathbf{C}_j\left(\mathbf{D}_{j+1}\mathbf{E}_j - \mathbf{D}_j\mathbf{E}_{j+1}\right), \qquad (2.131)\\
&= \mathbf{D}_j\left(\mathbf{C}_{j-1}\mathbf{E}_{j-1} + \mathbf{C}_j\mathbf{E}_{j+1}\right) - \left(\mathbf{D}_{j+1}\mathbf{C}_j + \mathbf{D}_{j-1}\mathbf{C}_{j-1}\right)\mathbf{E}_j,\\
&= \mathbf{D}_j\mathbf{H}_j\mathbf{E}_j - \mathbf{D}_j\mathbf{H}_j\mathbf{E}_j,\\
&= 0,
\end{aligned}$$

where we used the fact that the $\mathbf{C}_j$ are a constant times the identity (then they commute with all the matrices) and the recurrences satisfied by $\mathbf{D}_j$ and $\mathbf{E}_j$. □

Proposition 6. *The inverse of $\mathbf{H}$ is given by*

$$\mathbf{H}^{-1}_{j,k} = \begin{cases} -\mathbf{D}^t_j\mathbf{R}^{-t}\mathbf{E}^t_k & \text{if } j \le k,\\ -\mathbf{E}_j\mathbf{R}^{-1}\mathbf{D}_k & \text{if } j \ge k. \end{cases} \qquad (2.132)$$

Proof. It is a direct application of Proposition 2.4 in [17]. □

Proposition 7. *We have*

$$\left[\left(\mathbf{H}^{-1}_{j+1,j+1}\right)^{-1}\mathbf{H}^{-1}_{j+1,j}\right]\mathbf{H}^{-1}_{j+1,j} = \mathbf{H}^{-1}_{j,j} + \mathbf{C}^{-1}_{j+1}\left[\left(\mathbf{H}^{-1}_{j+1,j+1}\right)^{-1}\mathbf{H}^{-1}_{j+1,j}\right], \tag{2.133}$$

and

$$\left[\left(\mathbf{H}^{-1}_{j-1,j-1}\right)^{-1}\mathbf{H}^{-1}_{j-1,j}\right]\mathbf{H}^{-1}_{j-1,j} = \mathbf{H}^{-1}_{j,j} + \mathbf{C}^{-1}_{j-1}\left[\left(\mathbf{H}^{-1}_{j-1,j-1}\right)^{-1}\mathbf{H}^{-1}_{j-1,j}\right]. \tag{2.134}$$

Proof. From Eq. 2.132 we have

$$\left(\mathbf{H}^{-1}_{j+1,j+1}\right)^{-1}\mathbf{H}^{-1}_{j+1,j} = \mathbf{D}^{-1}_{j+1}\mathbf{D}_i. \tag{2.135}$$

Then we can compute

$$\begin{aligned}
\left[\left(\mathbf{H}^{-1}_{j+1,j+1}\right)^{-1}\mathbf{H}^{-1}_{j+1,j}\right]\mathbf{H}^{-1}_{j+1,j} &= -\mathbf{D}^{-1}_{j+1}\mathbf{D}_j\mathbf{E}_{j+1}\mathbf{R}^{-1}\mathbf{D}_j, && (2.136)\\
&= -\mathbf{E}_j\mathbf{R}^{-1}\mathbf{D}_j - \mathbf{D}^{-1}_{j+1}\mathbf{D}_j\left(\mathbf{E}_{j+1} - \mathbf{D}^{-1}_j\mathbf{D}_{j+1}\mathbf{E}_j\right)\mathbf{R}^{-1}\mathbf{D}_j, && (2.137)\\
&= -\mathbf{E}_j\mathbf{R}^{-1}\mathbf{D}_j - \mathbf{D}^{-1}_{j+1}\mathbf{D}_j\mathbf{D}^{-1}_j\left(\mathbf{D}_j\mathbf{E}_{j+1} - \mathbf{D}_{j+1}\mathbf{E}_j\right)\mathbf{R}^{-1}\mathbf{D}_j, && (2.138)\\
&= -\mathbf{E}_j\mathbf{R}^{-1}\mathbf{D}_j + \mathbf{D}^{-1}_{j+1}\mathbf{D}_j\mathbf{D}^{-1}_j\mathbf{C}^{-1}_j\mathbf{R}_{j+1}\mathbf{R}^{-1}\mathbf{D}_j, && (2.139)\\
&= -\mathbf{E}_j\mathbf{R}^{-1}\mathbf{D}_j + \mathbf{C}^{-1}_j\mathbf{D}^{-1}_{j+1}\mathbf{D}_j, && (2.140)\\
&= \mathbf{H}^{-1}_{j,j} + \mathbf{C}^{-1}_j\left[\left(\mathbf{H}^{-1}_{j+1,j+1}\right)^{-1}\mathbf{H}^{-1}_{j+1,j}\right], && (2.141)
\end{aligned}$$

in which we used the fact that the generalized Casorati determinand is constant. □

2.C Properties of the Discrete Green's representation formula

Proof of Lemma 1

Proof. We carry out the proof in 1D. The extension to the 2D case is trivial because of the Eq. 2.109, which is the discrete Green's representation formula in 2D, that takes in account the boundary conditions of $\mathbf{u}$ and $\mathbf{G}^{\ell}$. Let H be the discrete Helmholtz operator in 1D, and let u be the solution of

$$(Hu)_i = f_i, \qquad \text{for } i \in \mathbb{Z},$$

Let $\Omega = \{1, ..., n\}$. We define the discrete inner product as

$$\langle u, v\rangle_\Omega = \sum_{i=1}^{n} u_i v_i,$$

and the Green's function G_i^k, such that

$$\left(HG^k\right)_i = \delta_i^k, \qquad \text{for } i \in \mathbb{Z}.$$

Following the discretization given in Eq 2.102, we can write

$$\langle u, HG^k\rangle_\Omega - \langle G^k, Hu\rangle_\Omega = \mathcal{G}_k^{\downarrow}(u_0, u_1) + \mathcal{G}_k^{\uparrow}(u_n, u_{n+1}), \qquad \text{if } k \in \mathbb{Z}. \tag{2.142}$$

Applying the properties of G_k and the fact that u is the solution, we have

$$\langle u, \delta^k\rangle_\Omega - \langle G^k, f\rangle_\Omega = \mathcal{G}_k^{\downarrow}(u_0, u_1) + \mathcal{G}_k^{\uparrow}(u_n, u_{n+1}), \qquad \text{if } k \in \mathbb{Z}. \tag{2.143}$$

Following Def. 3 and Def. 2, we have

$$u_k = \mathcal{G}_k^{\downarrow}(u_0, u_1) + \mathcal{G}_k^{\uparrow}(u_n, u_{n+1}) + \mathcal{N}_k f, \qquad \text{if } 1 \le k \le n. \tag{2.144}$$

If $k < 1$ or $k > n$, the formula given by Eq. 2.144 is still valid. However, we have

$$\langle u, \delta^k\rangle_\Omega = 0,$$

because the support of the Dirac's delta is outside the integration domain. We obtain

$$-\langle G^k, f\rangle_\Omega = \mathcal{G}_k^{\uparrow}(u_n, u_{n+1}) + \mathcal{G}_k^{\downarrow}(u_0, u_1), \qquad \text{if } k < 1 \text{ or } k > n, \tag{2.145}$$

thus,

$$0 = \mathcal{G}_k^{\uparrow}(u_n, u_{n+1}) + \mathcal{G}_k^{\downarrow}(u_0, u_1) + \mathcal{N}_k f, \qquad \text{if } k < 1 \text{ or } k > n. \tag{2.146}$$

Finally, the only property that we used from G_k is that is should satisfy

$$\left(HG^k\right)_i = \delta_i^k, \qquad \text{for } 0 \le l \le n+1.$$

We can then replace the global Green's function by local ones, and the results still hold, i.e.,

$$u_k = \mathcal{G}_k^{\uparrow,1}(u_n, u_{n+1}) + \mathcal{G}_k^{\downarrow,1}(u_0, u_1) + \mathcal{N}_k f, \qquad \text{if } 1 \le k \le n, \tag{2.147}$$

$$0 = \mathcal{G}_k^{\uparrow,1}(u_n, u_{n+1}) + \mathcal{G}_k^{\downarrow,1}(u_0, u_1) + \mathcal{N}_k f, \qquad \text{if } k < 1 \text{ or } k > n, \tag{2.148}$$

which finishes the proof.

□

Proof of Lemma 2

Proof. We carry out the proof for $k = 0$, the case for $k = n^\ell + 1$ is analogous. Let us fix ℓ. By definition

$$\mathcal{G}_1^{\downarrow,\ell}(\mathbf{u}_0^\ell, \mathbf{u}_1^\ell) + \mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^\ell, \mathbf{u}_{n^\ell+1}^\ell) + \mathcal{N}_1^\ell \mathbf{f}^\ell = \mathbf{u}_1^\ell, \tag{2.149}$$

to which we apply the extrapolator, obtaining

$$\mathcal{E}^{\uparrow}_{\ell-1,\ell}\mathcal{G}^{\downarrow,\ell}_{1}(\mathbf{u}^{\ell}_{0},\mathbf{u}^{\ell}_{1}) + \mathcal{E}^{\uparrow}_{\ell-1,\ell}\mathcal{G}^{\uparrow,\ell}_{1}(\mathbf{u}^{\ell}_{n^{\ell}},\mathbf{u}^{\ell}_{n^{\ell}+1}) + \mathcal{E}^{\uparrow}_{\ell-1,\ell}\mathcal{N}^{\ell}_{1}\mathbf{f}^{\ell} = \mathcal{E}^{\uparrow}_{\ell-1,\ell}\mathbf{u}^{\ell}_{1}. \tag{2.150}$$

We compute each component of the last equation to show that it is equivalent to Eq. 2.42. Indeed, we use the definition of the extrapolator (Def. 6) and Lemma 4 to show that,

$$\mathcal{E}^{\uparrow}_{\ell-1,\ell}\mathcal{G}^{\uparrow,\ell}_{1}(\mathbf{u}^{\ell}_{n^{\ell}},\mathbf{u}^{\ell}_{n^{\ell}+1}) = \mathcal{G}^{\uparrow,\ell}_{0}(\mathbf{u}^{\ell}_{n^{\ell}},\mathbf{u}^{\ell}_{n^{\ell}+1}), \text{ and } \qquad \mathcal{E}^{\uparrow}_{\ell-1,\ell}\mathcal{N}^{\ell}_{1}\mathbf{f}^{\ell} = \mathcal{N}^{\ell}_{0}\mathbf{f}^{\ell}. \tag{2.151}$$

We compute the last term left using the matrix form of $\mathcal{G}^{\ell}$ (Eq. 2.24) which results in

$$\mathcal{E}^{\uparrow}_{\ell-1,\ell}\mathcal{G}^{\downarrow,\ell}_{1}(\mathbf{u}^{\ell}_{0},\mathbf{u}^{\ell}_{1}) = \frac{\mathcal{E}^{\uparrow}_{\ell-1,\ell}}{h}\left[\begin{array}{cc}\mathbf{G}^{\ell}(z_1,z_1) & -\mathbf{G}^{\ell}(z_1,z_0)\end{array}\right]\left(\begin{array}{c}\mathbf{u}^{\ell}_{0}\\ \mathbf{u}^{\ell}_{1}\end{array}\right). \tag{2.152}$$

Moreover, by direct application of Lemma 4 we have that

$$\mathcal{E}^{\uparrow}_{\ell-1,\ell}\mathbf{G}^{\ell}(z_1,z_1) = \mathbf{G}^{\ell}(z_0,z_1), \qquad \mathcal{E}^{\uparrow}_{\ell-1,\ell}\mathbf{G}^{\ell}(z_1,z_0) = \mathbf{G}^{\ell}(z_0,z_0) - h\mathcal{E}^{\uparrow}_{\ell-1,\ell}; \tag{2.153}$$

thus

$$\mathcal{E}^{\uparrow}_{\ell-1,\ell}\mathcal{G}^{\downarrow,\ell}_{1}(\mathbf{u}^{\ell}_{0},\mathbf{u}^{\ell}_{1}) = \mathcal{G}^{\downarrow,\ell}_{0}(\mathbf{u}^{\ell}_{0},\mathbf{u}^{\ell}_{1}) - \mathcal{E}^{\uparrow}_{\ell-1,\ell}\mathbf{u}^{\ell}_{1}. \tag{2.154}$$

Putting everything together we have that

$$\mathcal{G}^{\downarrow,\ell}_{0}(\mathbf{u}^{\ell}_{0},\mathbf{u}^{\ell}_{1}) + \mathcal{G}^{\uparrow,\ell}_{0}(\mathbf{u}^{\ell}_{n^{\ell}},\mathbf{u}^{\ell}_{n^{\ell}+1}) + \mathcal{E}^{\uparrow}_{\ell-1,\ell}\mathbf{u}^{\ell}_{1} + \mathcal{N}^{\ell}_{0}\mathbf{f}^{\ell} = \mathcal{E}^{\uparrow}_{\ell-1,\ell}\mathbf{u}^{\ell}_{1}, \tag{2.155}$$

or

$$\mathcal{G}^{\downarrow,\ell}_{0}(\mathbf{u}^{\ell}_{0},\mathbf{u}^{\ell}_{1}) + \mathcal{G}^{\uparrow,\ell}_{0}(\mathbf{u}^{\ell}_{n^{\ell}},\mathbf{u}^{\ell}_{n^{\ell}+1}) + \mathcal{N}^{\ell}_{0}\mathbf{f}^{\ell} = 0, \tag{2.156}$$

which concludes the proof. □

Proof of Lemma 3

Proof. The proof is a direct application of the nullity theorem [132]. If $\mathbf{u}^{\uparrow}$ and $\mathbf{v}^{\uparrow}$ are in the kernel of $\mathcal{A}^{\uparrow}_{j,j+1}$, then the proof is reduced to showing that $\mathbf{G}^{\ell+1}(z_1,z_1)$ is invertible. Without loss of generality we can reorder the entries of the matrix $\mathbf{G}^{\ell+1}$ such that $\mathbf{G}^{\ell+1}(z_1,z_1)$ is a square diagonal block located at the left upper corner of $\mathbf{G}^{\ell+1}$. Then the nullity of $\mathbf{G}^{\ell+1}(z_1,z_1)$ is equal to the nullity of the complementary block of the inverse, but the inverse is just the Helmholtz matrix reordered with some entries out. Such block is trivially full rank, i.e. nullity equals to zero. Then the nullity of $\mathbf{G}^{\ell+1}(z_1,z_1)$ is zero; therefore, $\mathbf{G}^{\ell+1}(z_1,z_1)$ is an invertible matrix. □

Proof of Lemma 4

Proof. We note that by the definition of the local Green's functions in Eq. 2.17, we have that

$$\mathbf{G}^{\ell}_{i,j,i',j'} = \frac{1}{h^2}\left(\mathbf{H}^{\ell}\right)^{-1}_{i,j,i',j'}. \tag{2.157}$$

Then by Def. 1 and Eq. 2.121 we have that

$$\mathbf{G}^{\ell}_{j,k} = \frac{1}{h}\left(\mathbf{H}^{\ell}\right)^{-1}_{j,k}, \tag{2.158}$$

where $\mathbf{G}^{\ell}_{j,k}$ is the layer to layer Green's function. Using the definition of the extrapolator (Def. 6) and Proposition 4, in particular Eq. 2.126 applied to $j = 0$ we obtain

$$\mathbf{G}^{\ell}(z_0, z_k) = \mathcal{E}^{\uparrow}_{\ell-1,\ell}\mathbf{G}^{\ell}(z_1, z_k), \qquad \text{for } 0 < k, \tag{2.159}$$

and Eq. 2.125 applied to $j = n^{\ell} + 1$

$$\mathbf{G}^{\ell}(z_{n^{\ell}+1}, z_k) = \mathcal{E}^{\downarrow}_{\ell,\ell+1}\mathbf{G}^{\ell}(z_{n^{\ell}}, z_k), \qquad \text{for } k < n^{\ell} + 1. \tag{2.160}$$

We can divide Eq. 2.134 by h, and we use the definitions of the extrapolator (Def. 6) and the Green's functions (Eq. 2.158) to obtain

$$\mathbf{G}^{\ell+1}(z_0, z_0) + \frac{\mathbf{C}_2^{-1}\mathcal{E}^{\uparrow}_{\ell,\ell+1}}{h} = \mathcal{E}^{\uparrow}_{\ell,\ell+1}\mathbf{G}^{\ell+1}(z_1, z_0). \tag{2.161}$$

However, following the notation of Eq. 2.121, we note that if k is such that it corresponds to a z_k that is in the physical domain, then $\mathbf{C}_k$ is just $-I/h^2$. This observation is independent of the formulation, and it is due to the particular ordering of the unknowns that Eq. 2.121 assumes. Then we can further simplify Eq. 2.161 and obtain

$$\mathbf{G}^{\ell+1}(z_0, z_0) - h\mathcal{E}^{\uparrow}_{\ell,\ell+1} = \mathcal{E}^{\uparrow}_{\ell,\ell+1}\mathbf{G}^{\ell+1}(z_1, z_0). \tag{2.162}$$

We can follow the same reasoning to obtain from Eq. 2.133 that

$$\mathbf{G}^{\ell}(z_{n^{\ell}+1}, z_{n^{\ell}+1}) - h\mathcal{E}^{\downarrow}_{\ell,\ell+1} = \mathcal{E}^{\downarrow}_{\ell,\ell+1}\mathbf{G}^{\ell}(z_{n^{\ell}}, z_{n^{\ell}+1}), \tag{2.163}$$

which concludes the proof.

□

Proof of Lemma 5

Proof. We have that

$$\mathcal{G}^{\downarrow,\ell+1}_1(\mathbf{u}^{\uparrow}_0, \mathbf{u}^{\uparrow}_1) = 0 \tag{2.164}$$

is, by definition, equivalent to

$$\mathbf{G}^{\ell+1}(z_1, z_1)\mathbf{u}^{\uparrow}_0 = \mathbf{G}^{\ell+1}(z_1, z_0)\mathbf{u}^{\uparrow}. \tag{2.165}$$

The proof is by induction, at each stage we use the extrapolator to shift the evaluation index. We left multiply Eq. 2.165 by $\left[\mathbf{G}^{\ell+1}(z_1, z_1)\right]^{-1}\mathbf{G}^{\ell+1}(z_1, z_2)$ and follow Remark 2, to obtain

$$\mathbf{G}^{\ell+1}(z_2, z_1)\mathbf{u}^{\uparrow}_0 = \mathbf{G}^{\ell+1}(z_2, z_0)\mathbf{u}^{\uparrow}, \tag{2.166}$$

which can be left multiplied by the matrix $\left[\mathbf{G}^{\ell+1}(z_2, z_2)\right]^{-1} \mathbf{G}^{\ell+1}(z_2, z_3)$ to obtain

$$\mathbf{G}^{\ell+1}(z_3, z_1)\mathbf{u}_0^{\uparrow} = \mathbf{G}^{\ell+1}(z_3, z_0)\mathbf{u}^{\uparrow}. \tag{2.167}$$

Then by induction we obtain the result. □

Proof of Lemma 6

Proof. We give the proof for the case when $j = 0$ – for $j = n^\ell + 1$ the proof is analogous.

Given that $\underline{\underline{\mathbf{u}}}$ is solution of the system in Def. 8, we have

$$\mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow}, \mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_1^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow}, \mathbf{u}_1^{\ell,\downarrow}) + \mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\downarrow}, \mathbf{u}_{n^\ell+1}^{\ell,\downarrow}) + \mathcal{N}_1^\ell \mathbf{f}^\ell = \mathbf{u}_1^{\ell,\uparrow} + \mathbf{u}_1^{\ell,\downarrow}, \tag{2.168}$$

which can be left-multiplied by the extrapolator $\mathcal{E}_{\ell,\ell+1}^{\uparrow}$. Using Lemma 4 we have

$$\mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow}, \mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{E}_{\ell,\ell+1}^{\uparrow}\mathcal{G}_1^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow}, \mathbf{u}_1^{\ell,\downarrow}) + \mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\downarrow}, \mathbf{u}_{n^\ell+1}^{\ell,\downarrow}) + \mathcal{N}_1^\ell \mathbf{f}^\ell = \mathcal{E}_{\ell,\ell+1}^{\uparrow}\mathbf{u}_1^{\ell,\uparrow} + \mathcal{E}_{\ell,\ell+1}^{\uparrow}\mathbf{u}_1^{\ell,\downarrow}, \tag{2.169}$$

and following the same computation performed in Lemma 2 (Eq. 2.154) we have

$$\mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow}, \mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_0^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow}, \mathbf{u}_1^{\ell,\downarrow}) + \mathcal{E}_{\ell,\ell+1}^{\uparrow}\mathbf{u}_1^{\ell,\downarrow} + \mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\downarrow}, \mathbf{u}_{n^\ell+1}^{\ell,\downarrow}) + \mathcal{N}_0^\ell \mathbf{f}^\ell = \mathcal{E}_{\ell,\ell+1}^{\uparrow}\mathbf{u}_1^{\ell,\uparrow} + \mathcal{E}_{\ell,\ell+1}^{\uparrow}\mathbf{u}_1^{\ell,\downarrow}. \tag{2.170}$$

Finally, from the fact that $\mathcal{E}_{\ell,\ell+1}^{\uparrow}\mathbf{u}_1^{\ell,\uparrow} = \mathbf{u}_0^{\ell,\uparrow}$ we obtain that

$$\mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow}, \mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_0^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow}, \mathbf{u}_1^{\ell,\downarrow}) + \mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\downarrow}, \mathbf{u}_{n^\ell+1}^{\ell,\downarrow}) + \mathcal{N}_0^\ell \mathbf{f}^\ell = \mathbf{u}_0^{\ell,\uparrow}. \tag{2.171}$$

□

Proof of Proposition 1

Proof. The sufficient condition is given by Lemma 6, so we focus on the necessary condition. The proof can be reduced to showing that

$$\mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow}, \mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_0^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow}, \mathbf{u}_1^{\ell,\downarrow}) + \mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\downarrow}, \mathbf{u}_{n^\ell+1}^{\ell,\downarrow}) + \mathcal{N}_0^\ell \mathbf{f}^\ell = \mathbf{u}_0^{\ell,\uparrow}, \tag{2.172}$$

implies $\mathcal{E}_{\ell,\ell+1}^{\uparrow}\mathbf{u}_1^{\ell,\uparrow} = \mathbf{u}_0^{\ell,\uparrow}$. Indeed, we have

$$\mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow}, \mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_0^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow}, \mathbf{u}_1^{\ell,\downarrow}) + \mathcal{E}_{\ell,\ell+1}^{\uparrow}\mathbf{u}_1^{\ell,\downarrow} + \mathcal{G}_0^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\downarrow}, \mathbf{u}_{n^\ell+1}^{\ell,\downarrow}) + \mathcal{N}_0^\ell \mathbf{f}^\ell = \mathbf{u}_0^{\ell,\uparrow} + \mathcal{E}_{\ell,\ell+1}^{\uparrow}\mathbf{u}_1^{\ell,\downarrow}. \tag{2.173}$$

Given that the extrapolator is invertible (Proposition 7 and Remark 3) we can multiply the equation above on the left by $\left(\mathcal{E}_{\ell,\ell+1}^{\uparrow}\right)^{-1}$. Using the same computations performed in Lemma 2 we have that

$$\mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow}, \mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_1^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow}, \mathbf{u}_1^{\ell,\downarrow}) + \mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\downarrow}, \mathbf{u}_{n^\ell+1}^{\ell,\downarrow}) + \mathcal{N}_1^\ell \mathbf{f}^\ell = \left(\mathcal{E}_{\ell,\ell+1}^{\uparrow}\right)^{-1} \mathbf{u}_0^{\ell,\uparrow} + \mathbf{u}_1^{\ell,\downarrow}. \tag{2.174}$$

Moreover, by hypothesis $\underline{\underline{\mathbf{u}}}$ satisfies

$$\mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\uparrow}, \mathbf{u}_{n^\ell+1}^{\ell,\uparrow}) + \mathcal{G}_1^{\downarrow,\ell}(\mathbf{u}_0^{\ell,\downarrow}, \mathbf{u}_1^{\ell,\downarrow}) + \mathcal{G}_1^{\uparrow,\ell}(\mathbf{u}_{n^\ell}^{\ell,\downarrow}, \mathbf{u}_{n^\ell+1}^{\ell,\downarrow}) + \mathcal{N}_1^\ell \mathbf{f}^\ell = \mathbf{u}_1^{\ell,\uparrow} + \mathbf{u}_1^{\ell,\downarrow}, \tag{2.175}$$

which simplifies to

$$\mathbf{u}_1^{\ell,\uparrow} = \left(\mathcal{E}_{\ell,\ell+1}^{\uparrow}\right)^{-1} \mathbf{u}_0^{\ell,\uparrow}. \tag{2.176}$$

Finally, using the fact that the extrapolator is invertible, we obtain the desired result. The proof for $i = n^\ell$ is analogous.

□

Proof of Theorem 1

Proof. Once again we start by proving the statement in 1D. One side of the equivalence is already given by Lemma 1. To show the other side, we need to show that the concatenated solution satisfy the solution at every point.

Let $\underline{\mathbf{u}}$ be the solution to the discrete integral equation. We can then reconstruct the local solution at each subdomain by

$$u_k^\ell = \mathcal{G}_k^{\uparrow,\ell}(u_{n^\ell}^\ell, u_{n^\ell+1}^\ell) + \mathcal{G}_k^{\downarrow,\ell}(u_0^\ell, u_1^\ell) + \mathcal{N}_k^\ell f^\ell. \tag{2.177}$$

We have

$$\left(Hu^\ell\right)_i = f_i^\ell,$$

for $1 < l < N^\ell$. To conclude we need to prove that the difference equation is satisfied at $i = 1$ and at $i = n^\ell$ with the information coming from the neighboring sub-domains. We remark that Eq. 2.177 is equivalent to solving

$$\left(Hu^\ell\right)_i = f_i^\ell - \delta_{0,l}(\partial_x^+ u_0^\ell) + (\partial_x^+ \delta_{0,l}) u_0^\ell + \delta_{n^\ell+1,i}(\partial_x^+ u_{n^\ell+1}^\ell) - (\partial_x^- \delta_{N^\ell+1,i}) u_{n^\ell+1}^\ell \tag{2.178}$$

where

$$\delta_{i,j} = \begin{cases} \frac{1}{h} & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{2.179}$$

and the up and down-wind derivatives were defined earlier. Using the fact that $u_i^\ell = 0$ if $i = 0$ or $i = n^\ell$ (Eq. 2.42), and the equivalence between the Green's representation formula and the problem stated in Eq. 2.178, we can apply H to u^ℓ and evaluate it at $i = 1$ obtaining,

$$\left(Hu^\ell\right)_1 = \frac{2u_1^\ell - u_2^\ell}{h^2} - m_1^\ell \omega^2 u_1^\ell = f_1^\ell + \frac{\delta_{1,l}}{h} u_0^\ell = f_1^\ell + \frac{u_0^\ell}{h^2}. \tag{2.180}$$

In other words,

$$\frac{-u^{\ell}_0 + 2u^{\ell}_1 - u^{\ell}_2}{h^2} - m^{\ell}_1 \omega^2 u^{\ell}_1 = f^{\ell}_1, \tag{2.181}$$

and by construction $u^{\ell}_0 = u^{\ell-1}_{n^{\ell-1}}$. This procedure can be replicated for $i = n^{\ell}$, yielding

$$\frac{-u^{\ell}_{n^{\ell}-1} + 2u^{\ell}_{n^{\ell}} - u^{\ell}_{n^{\ell}+1}}{h^2} - m^{\ell}_{n^{\ell}} \omega^2 u^{\ell}_{n^{\ell}} = f^{\ell}_{n^{\ell}}, \tag{2.182}$$

in which we have $u^{\ell}_{n^{\ell}+1} = u^{\ell+1}_1$. This means that the concatenated solution satisfies the equation at the interfaces; therefore, it satisfies the difference equation everywhere. By construction it satisfies the boundary conditions; therefore, by uniqueness it is the solution to the difference equation.

In 2D, we need to be especially careful with the PML. Using the same proof method as before, we have that

$$\mathbf{u}^{\ell}_k = \mathcal{G}^{\uparrow,\ell}_k(\mathbf{u}^{\ell}_{n^{\ell}}, \mathbf{u}^{\ell}_{n^{\ell}+1}) + \mathcal{G}^{\downarrow,\ell}_k(\mathbf{u}^{\ell}_0, \mathbf{u}^{\ell}_1) + \mathcal{N}^{\ell}_k \mathbf{f}^{\ell}, \tag{2.183}$$

satisfies the equation

$$\left(\mathbf{H}\mathbf{u}^{\ell}\right)_{i,j} = \mathbf{f}^{\ell}_{i,j}, \qquad \text{for } 1 < j < n^{\ell} \text{ and} - n_{\text{pml}} + 1 < i < n^{\ell} + n_{\text{pml}}$$

where i and j are local indices. Following the same reasoning as in the 1D case we have

$$\left(\mathbf{H}\mathbf{u}^{\ell}\right)_{i,1} = \mathbf{f}^{\ell}_{i,1} - \frac{1}{h^2}\mathbf{u}^{\ell}_{i,0}, \qquad \text{for and} - n_{\text{pml}} + 1 < i < n^{\ell} + n_{\text{pml}}. \tag{2.184}$$

To prove Eq. 2.184 we use the fact that $\mathbf{u}^{\ell}_k$ is defined by Eq. 2.183, and by Lemma 2, $\mathbf{u}^{\ell}_k = 0$ for $k = 0$ and $k = n^{\ell} + 1$. Then, if we apply the global finite differences operator, $\mathbf{H}$, to the local $\mathbf{u}^{\ell}_k$, and to evaluate it at $k = 1$, we obtain

$$\begin{aligned}
\left(\mathbf{H}\mathbf{u}^{\ell}\right)_{i,1} = & -\alpha_x(\mathbf{x}_{i,1}) \frac{\alpha_x(\mathbf{x}_{i+1/2,1})(\mathbf{u}^{\ell}_{i+1,1} - \mathbf{u}^{\ell}_{i,1}) - \alpha_x(\mathbf{x}_{i-1/2,1})(\mathbf{u}^{\ell}_{i,1} - \mathbf{u}^{\ell}_{i-1,1})}{h^2} \\
& + \frac{1}{h^2}\left(2\mathbf{u}^{\ell}_{i,1} - \mathbf{u}^{\ell}_{i,2}\right) - \omega^2 m(\mathbf{x}_{i,1}) \qquad (2.185)\\
= & \, \mathbf{f}^{\ell}_{i,1} + \frac{1}{h^2}\mathbf{u}^{\ell}_{i,0}. \qquad (2.186)
\end{aligned}$$

It is clear that the right-hand side has a similar form as in the 1D case. The concatenated solution satisfies the discretized PDE at the interfaces. Moreover, we can observe that by construction $\mathbf{u}^{\ell}$ satisfies the homogeneous Dirichlet boundary conditions, because the Green's functions satisfy the same boundary conditions. Furthermore, the traces at the interface also satisfy the zero boundary conditions at the endpoints. Then, the concatenated solution satisfy the finite difference equation inside the domain and the boundary conditions; therefore, by uniqueness it is solution to the discrete PDE. □

Chapter 3

Extensions

The method of polarized traces introduced in Chapter 2 is an efficient and scalable iterative solver for the Helmholtz equation, and constitutes a new approach to domain decomposition for high-frequency wave propagation. Its online runtime is $\mathcal{O}(N/P)$, provided that $P = \mathcal{O}(N^{1/8})$, where N is the number of degrees of freedom and P the number of processors in a distributed memory environment. However, it has its own limitations, in particular:

- the complexity degrades greatly when the medium exhibits internal cavities and sharp contrasts,
- the offline precomputation, which involves computing, storing and compressing interface-to-interface Green's functions, can become prohibitively computationally expensive for large problems.

In this chapter we introduce new ideas to mitigate these issues at some extent. The main results are

- a compressed-block LU solver based on the discrete boundary integral formulation coupled with $\mathcal{H}$-matrices, whose online runtimes are not degraded by the presence of internal cavities or sharp contrasts, at the expense of a more thorough precomputation;
- a nested domain decomposition approach that allows us to reduce the offline and online costs. At the inner level of the nested decomposition an efficient solver is needed, the best choice seems to be the compressed-block LU solver, even in the absence of resonant cavities. In this case the cost of the precomputation for the compressed-block LU solver remains acceptable. The resulting algorithm has an asymptotic online runtime of $\mathcal{O}(N/P)$ provided that $P = \mathcal{O}(N^{1/5})$, which results in a lower online runtime in a distributed memory environment.

In addition, we propose a few improvements to the original scheme presented in Chapter 2 to obtain better accuracy, and to accelerate the convergence rate. We provide :

- an equivalent formulation of the method of polarized traces that involves discretizations using Q1 finite elements, which is second order accurate despite the roughness of the model, provided that a suitable quadrature rule is used to compute the mass matrix. This formulation can be easily generalized for high-order finite differences and high-order finite elements;
- and, a variant of the preconditioner introduced in Chapter 2, in which we used a block Gauss-Seidel iteration instead of a block Jacobi iteration, that improves the convergence rate.

Organization

The present chapter is organized as follows :

- we review briefly the formulation of the Helmholtz problem and the reduction to a boundary integral equation in Section 3.1;
- in Section 3.2 we present the compressed-block LU solver, and provide the empirical complexities;
- in Section 3.3 we present the nested solver, we introduce the two variants, one of which relies on the compressed-block LU solver at the inner level, and we provide the empirical complexity observed;
- finally, in Section 3.4 we present numerical experiments that corroborate the complexity claims.

3.1 Formulation

As in Chapter 2, let Ω be a rectangle in $\mathbb{R}^2$, and consider a layered partition of Ω into L slabs, or layers $\{\Omega^\ell\}_{\ell=1}^L$. Define the squared slowness as $m(\mathbf{x}) = 1/c(\mathbf{x})^2$, $\mathbf{x} = (x, z)$. Define the global Helmholtz operator at frequency ω as

$$\mathcal{H}u = \left(-\triangle - m\omega^2\right) u \qquad \text{in } \Omega, \tag{3.1}$$

with an absorbing boundary condition on $\partial\Omega$.

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

$$\mathcal{H}^\ell u = \left(-\triangle - m\omega^2\right) u \qquad \text{in } \Omega^\ell, \tag{3.2}$$

with an absorbing boundary condition on $\partial\Omega^\ell$. Let u be the solution to $\mathcal{H}u = f$.

Following Section 2.2, after discretization we solve the linear system

$$\mathbf{H}\mathbf{u} = \mathbf{f}, \tag{3.3}$$

for the global solution, and

$$\mathbf{H}^{\ell}\mathbf{u}^{\ell} = \mathbf{f}^{\ell}, \tag{3.4}$$

for the local ones. In this case we use a slightly different strategy to build the absorbing boundary conditions at the interfaces; instead of using a normal extension as in Section 2.2 to build the absorbing layer, we use the true wave speed.

We perform the same reduction to a discrete boundary integral system as in Chapter 2. Solving the problem in the volume is equivalent to solving

$$\underline{\mathbf{M}}\underline{\mathbf{u}} = \underline{\mathbf{f}}, \tag{3.5}$$

at the interfaces between slabs. Once the trace of the solution at the interfaces, $\underline{\mathbf{u}}$, is known, we reconstruct locally the solution at each layer using the Green's representation formula, operation that is completely parallel.

We point out that the number of layers L is different for each solver, for the layered partition inherent to the compressed-block LU we assume that $L = P$; whereas for the nested solve we suppose that $L \sim P^{1/2}$, and each slab is subdivided in $L_c \sim P^{1/2}$ cells such that $LL_c = P$.

3.2 A compressed-block LU solver

The method of polarized traces presented in Chapter 2 is a highly efficient solver for the Helmholtz equation; however, its performance often degrades when dealing with media featuring resonant cavities and sharp contrasts. This adverse effect is common to all iterative methods, and is mainly due to the underlying physics of the problem. Iterative methods can handle waves propagating through the domain efficiently, but they can only capture waves that were scattered k times during the first k iterations. Physically, the scattered waves are converted into waves propagating in a different direction, which are handled in a posterior iteration. In the case of a large cavity, the solution of the Helmholtz equation is composed of the superposition of several waves reflecting from the internal interfaces of the cavity, which dramatically increases the number of iterations needed for convergence.

In general, for the case of large resonant cavities, direct methods are used [88].

We explore a variant of the method of polarized traces, using a layered domain decomposition, that yields, surprisingly, an identical empirical online parallel complexity than the one observed for the method of polarized traces in Chapter 2. As stated before, its online complexity does not seem to deteriorate in the presence of large resonant cavities and sharp interfaces; however, the method has a more thorough offline precomputation.

The method may be called "compressed-block LU solver". Its ingredients (such as adaptive $\mathcal{H}$-matrices) are not particularly novel by themselves, but the documentation of the online complexity claim seems to be new in the case of large resonant cavities. The method is potentially attractive in situations where the precomputation is amortized over many right-hand sides. Such applications range from optimizing the shape of waveguides and ring resonators in nanophotonics, to optimal survey design for seismic prospection, and to optimal focusing for intra-craneal treatments using high intensity ultrasound.

The method consists in a block LU factorization without pivoting of $\underline{\mathbf{M}}$, and a compression of the blocks of the LU factors in PLR form[1]. The compression of the blocks is unexpectedly good in the high-frequency regime, provided that an accurate discretization is used. There is no known theory, to the author's knowledge, that can fully explain this surprising behavior. We only provide numerical evidence of the scalings.

3.2.1 Method

Instead of solving $\underline{\mathbf{Mu}} = \underline{\mathbf{f}}$ (Eq. 3.5) using the method of polarized traces, i.e., using a iterative method, we perform a direct solve. Given the banded structure of $\underline{\mathbf{M}}$ (see Fig. 3-1) a LU factorization without pivoting will preserve its sparsity pattern.

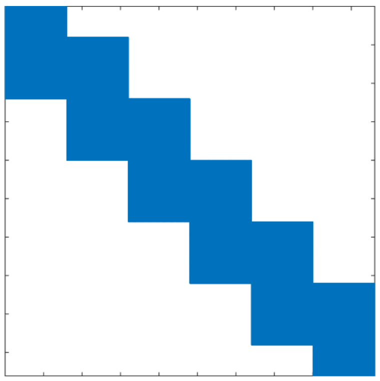


Figure 3-1: Sparsity pattern of Eq. 3.5.

Given that $\underline{\mathbf{M}}$ is distributed among different processors, we perform a block LU decomposition

$$\underline{\mathbf{L}}\,\underline{\mathbf{U}} = \underline{\mathbf{M}}. \tag{3.6}$$

The sparsity pattern of the LU factors are depicted in Fig. 3-2. Furthermore, the diagonal blocks of the LU factors are inverted explicitly so that the forward and backward substitutions are reduced to a series of matrix-vector products.

Finally, the blocks are compressed in PLR form, yielding a fast solve.

[1] PLR matrices are $\mathcal{H}$-matrices with a dyadic partitioning and a fully adaptive admissibility condition based on the ϵ-rank of its blocks, for further details see Section 2.5.

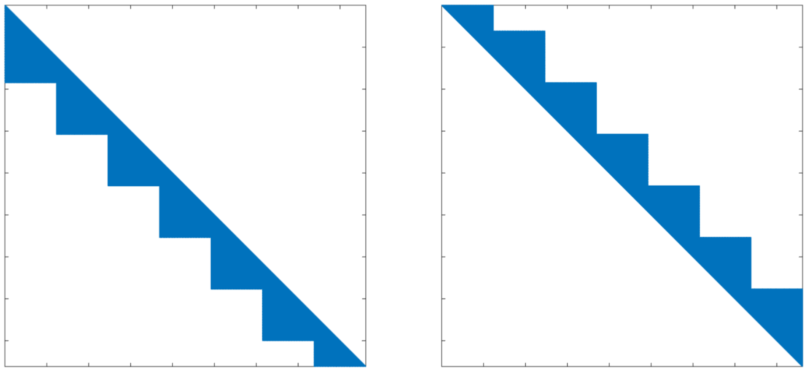


Figure 3-2: Sparsity pattern of the LU factorization in Eq 3.6.

We solve Eq. 2.15 using,

$$\underline{\mathbf{u}} = \left(\underline{\mathbf{U}}\right)^{-1}\left(\underline{\mathbf{L}}\right)^{-1}\underline{\mathbf{f}}. \tag{3.7}$$

This method is suitable to solve multi-right-hand sides simultaneously, using BLAS-3 operations (or sparse BLAS), which are highly efficient.

The resulting algorithm is presented in Alg. 7.

Algorithm 7. *Online computation for the compressed-block LU solver*

```
1:  function $\mathbf{u}$ = HELMHOLTZ SOLVER( $\mathbf{f}$ )
2:      for $\ell = 1 : L$ do
3:          $\mathbf{f}^\ell = \mathbf{f}\chi_{\Omega^\ell}$                          ▷ partition the source
4:      end for
5:      for $\ell = 1 : L$ do
6:          $\mathcal{N}^\ell \mathbf{f}^\ell = (\mathbf{H}^\ell)^{-1}\mathbf{f}^\ell$                  ▷ solve local problems
7:      end for
8:      $\underline{\mathbf{f}} = \left(\mathcal{N}^1_{n^1}\mathbf{f}^1, \mathcal{N}^2_1\mathbf{f}^2, \mathcal{N}^2_{n^2}\mathbf{f}^2, \dots, \mathcal{N}^L_1\mathbf{f}^L\right)^t$   ▷ form r.h.s. for the integral system
9:      $\underline{\mathbf{u}} = \left(\underline{\mathbf{U}}\right)^{-1}\left(\underline{\mathbf{L}}\right)^{-1}\underline{\mathbf{f}}$            ▷ solve for the traces (Eq. 3.7)
10:     for $\ell = 1 : L$ do
11:         $\mathbf{u}^\ell_j = \mathcal{G}^{\uparrow,\ell}_j(\mathbf{u}^\ell_{n^\ell}, \mathbf{u}^\ell_{n^\ell+1}) + \mathcal{G}^{\downarrow,\ell}_j(\mathbf{u}^\ell_0, \mathbf{u}^\ell_1) + \mathcal{N}^\ell_j\mathbf{f}^\ell$   ▷ reconstruct local solutions (Eq. 2.28)
12:     end for
13:     $\mathbf{u} = (\mathbf{u}^1, \mathbf{u}^2, \dots, \mathbf{u}^{L-1}, \mathbf{u}^L)^t$            ▷ concatenate the local solutions
14: end function
```

3.2.2 Complexity

We summarize the complexity of the compressed-block LU in Table 3.1, in which we suppose that $L = P$, i.e., we have one processor per layer. We suppose that each node has access to the wave speed $c(\mathbf{x})$ in the corresponding subdomain at the beginning of the precomputation. For the online stage, we assume that the source is known locally at each node at the beginning of the computation, and we suppose that solver has finished when the solution is reconstructed locally at each node.

The offline stage of the compressed direct solver is comprised of the LU factorization of the local problems, the computation of the local Green's functions to assemble $\underline{\mathbf{M}}$, its factorization, the inversion of the diagonal blocks of the LU factors of $\underline{\mathbf{M}}$, and the compression in PLR form of the blocks of the modified LU factors. The overall offline time complexity is dominated by the factorization of the discrete integral system, which is $\mathcal{O}(LN^{3/2})$.

In the cost analysis, we suppose that the LU factorization, the computation of the Green's functions, the inversion of the diagonal blocks and the compression of the blocks are performed locally at each layer, incurring no communication cost. The only communication cost arises from the block LU factorization, which needs to transfer a full block of size n^2, for each step of the block LU. This implies a communication cost of $\mathcal{O}(LN)$.

The online stage has an embarrassingly parallel stage, which is comprised of local solves at each layer ($\mathcal{O}(N/L)$) and the local reconstruction in the volume ($\mathcal{O}(N/L)$); and a sequential stage, which involves solving of Eq. 3.5 using Eq. 3.7. To solve Eq. 3.5 we perform $\mathcal{O}(L)$ compressed matrix-vector product sequentially. The complexity of the compressed matrix-vector product depends on the scaling of the frequency with respect to n and the accuracy of the discretization. For example, if the medium is smooth, second order finite differences are known to be accurate provided that the frequency scales as $\omega \sim \sqrt{n}$. However, in the presence of sharp interfaces, the accuracy is greatly degraded, which makes it difficult to obtain an asymptotic estimate for the compression in PLR form of the blocks of the LU factors. We circumvent this problem by discretizing Eq. 3.1 using Q1 finite elements with an adaptive quadrature. For further details see Appendices 3.A and 3.B.

For the communication cost of the online stage, there is no difference between this method and the method of polarized traces. The communication cost is $\mathcal{O}(LN^{1/2})$ or better.

If the frequency scales as $\omega \sim \sqrt{n}$, the regime in which second order Q1 finite elements are expected to be accurate (see [83]), we obtain empirically that $\alpha = 5/8$; however, we assume the more conservative value of $\alpha = 3/4$. The latter is in better agreement with a theoretical analysis of the rank of the off-diagonal blocks of the Green's functions for smooth wave speed. We point that the scaling obtained are surprisingly good, and there is no theory that fully explains the results obtained.

The total cost of solving the boundary integral system is $\mathcal{O}(LN^{3/4})$, which implies that the online cost is $\mathcal{O}(N/L + LN^{3/4})$. As in [148], the observed empirical online runtime is $\mathcal{O}(N/L)$ provided that $L = \mathcal{O}(N^{1/8})$.

3.3 Nested solver

The other main drawback of the method of polarized traces, as well as the compressed-block LU strategy presented earlier, is its offline precomputation that involves computing and storing the interface-to-interface Green's functions. 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

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

Table 3.1: Complexity of the different steps of the compressed-block LU. We suppose that we have one processor per layer, $L = P$. Typically $\alpha = 3/4$.

Step	N_{nodes}	Communication
LU factorizations	$\mathcal{O}(P)$	$\mathcal{O}\left(1\right)$
Green's functions	$\mathcal{O}(P)$	$\mathcal{O}\left(1\right)$
Block LU factorization	1	$\mathcal{O}\left(PN\right)$
Local solves	$\mathcal{O}(P)$	$\mathcal{O}\left(N^{1/2}\right)$
Sweeps	2	$\mathcal{O}(PN^{1/2})$
Recombination	$\mathcal{O}(P)$	$\mathcal{O}\left(N^{1/2}\right)$

Table 3.2: Communication cost of the different steps of the compressed-block LU. We suppose that we have one processor per layer, $L = P$.

with sources at the interfaces between layers.

In particular, the method of polarized traces relies on solving the polarized integral system (the default formulation, which relies on the jump conditions)

$$\underline{\underline{\mathbf{M}}}\,\underline{\underline{\mathbf{u}}} = \underline{\underline{\mathbf{f}}}, \qquad \underline{\underline{\mathbf{u}}} = \left(\begin{array}{c} \underline{\mathbf{u}}^{\downarrow} \\ \underline{\mathbf{u}}^{\uparrow} \end{array} \right). \tag{3.8}$$

The matrix $\underline{\underline{\mathbf{M}}}$ takes the form

$$\underline{\underline{\mathbf{M}}} = \left[\begin{array}{cc} \underline{\mathbf{D}}^{\downarrow} & \underline{\mathbf{U}} \\ \underline{\mathbf{L}} & \underline{\mathbf{D}}^{\uparrow} \end{array} \right], \tag{3.9}$$

where $\underline{\mathbf{D}}^{\downarrow}$ and $\underline{\mathbf{D}}^{\uparrow}$ are, respectively, block lower and upper diagonal matrices with identity diagonal blocks, thus easily invertible, using a block back-substitution. In the method of polarized traces, Eq. 3.8 is solved iteratively, using an efficient preconditioner that relies on the application of $\left(\underline{\mathbf{D}}^{\downarrow}\right)^{-1}$ and $\left(\underline{\mathbf{D}}^{\uparrow}\right)^{-1}$.

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 a restriction of local Green's functions. Thus they can be applied via a local solve (using, for example, a direct solver) with sources at the interfaces, which is the same argument we used in Chapter 2 to reconstruct the solution in the volume using Eq. 2.27. However, given the iterative nature of the preconditioner (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. 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.

Let P be the number of nodes in a distributed memory environment. If we suppose that each layer is associated with one node, then the method of polarized traces' online runtime is $\mathcal{O}(N/P)$ as long as $P = \mathcal{O}(N^{1/8})$. In this chapter we present a variant of the method of the polarized traces with improved complexity and lower memory footprint, with an online runtime $\mathcal{O}(N/P)$ provided that $P = \mathcal{O}(N^{1/5})$.

The nested domain decomposition 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. 3-3.

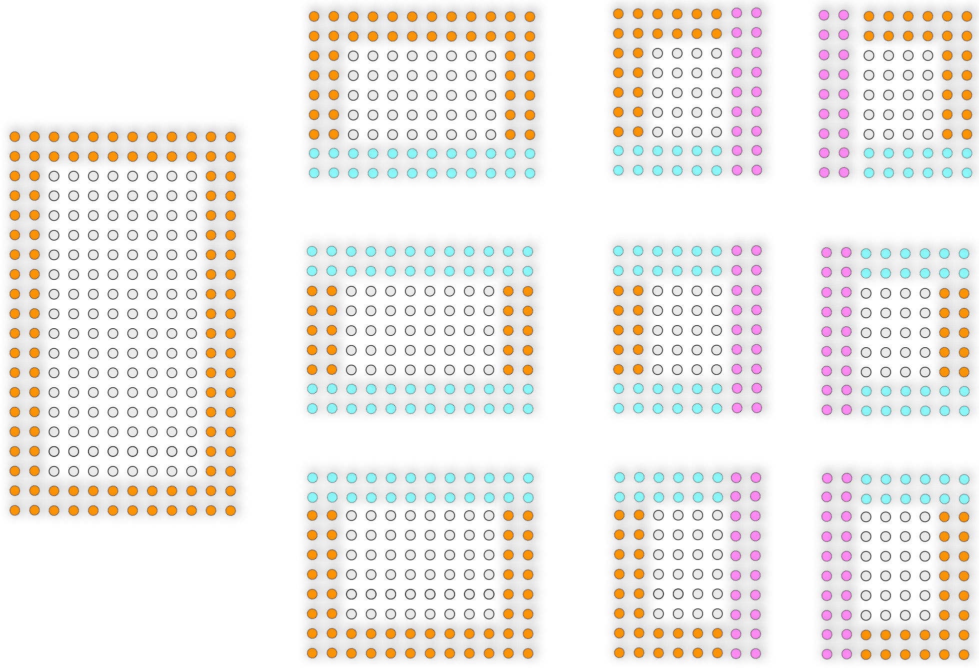


Figure 3-3: Nested Decomposition in cells. The orange grid-points represent the PML for the original problem, the light-blue represent the artificial PML between layers, and the pink grid-points represent the artificial PML between cells in the same layer.

Finally, 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 requirement.

The nested approach consists of two levels:

- the *outer* solver, which solves the global Helmholtz problem (Eq. 3.3), using the method of polarized traces to solve Eq. 3.5 at the interfaces between layers;

- and the *inner* solver, which solves the local Helmholtz problems at each layer (Eq. 3.4), using an integral boundary equation to solve for the degrees of freedom a the interfaces between cells within a layer.

3.3.1 Gauss-Seidel preconditioner

In this chapter, we use the block Gauss-Seidel iteration as a preconditioner to solve the outer polarized system in Eq. 2.41.

We define the Gauss-Seidel iteration in Alg. 8 following the notation in Chapter 2, and we give the expression of the preconditioner used later for the outer solver in Eq. 3.10. For the sake of clarity, we present the matrix version of the preconditioner in this Section; the algorithms for the matrix-free version are provided later in Section 3.3.2

Algorithm 8. *Gauss-Seidel iteration*

```
1: function $\underline{\mathbf{u}} = \text{Gauss-Seidel}(\underline{\underline{\mathbf{f}}}, \epsilon_{tol})$
2:     $\underline{\underline{\mathbf{u}}}^0 = (\underline{\mathbf{u}}^{\downarrow}, \underline{\mathbf{u}}^{\uparrow})^t = 0$
3:     while $\|\underline{\underline{\mathbf{u}}}^{n+1} - \underline{\underline{\mathbf{u}}}^n\| / \|\underline{\underline{\mathbf{u}}}^n\| > \epsilon_{tol}$ do
4:         $\begin{pmatrix} \underline{\mathbf{v}}^{\downarrow} \\ \underline{\mathbf{v}}^{\uparrow} \end{pmatrix} = \underline{\underline{\mathbf{f}}} - \begin{bmatrix} 0 & \underline{\mathbf{U}} \\ 0 & 0 \end{bmatrix} \underline{\underline{\mathbf{u}}}^n$
5:         $\underline{\underline{\mathbf{u}}}^{n+1} = \begin{pmatrix} (\underline{\mathbf{D}}^{\downarrow})^{-1}\underline{\mathbf{v}}^{\downarrow} \\ (\underline{\mathbf{D}}^{\uparrow})^{-1}\left(\mathbf{v}^{\uparrow} - \underline{\mathbf{L}}(\underline{\mathbf{D}}^{\downarrow})^{-1}\underline{\mathbf{v}}^{\downarrow}\right) \end{pmatrix}$
6:     end while
7:     $\underline{\mathbf{u}} = \underline{\mathbf{u}}^{\uparrow,n} + \underline{\mathbf{u}}^{\downarrow,n}$
8: end function
```

The Gauss-Seidel iteration yields faster convergence than the block Jacobi iteration in Chapter 2. In our experiments, using GMRES preconditioned with one Gauss-Seidel iteration converged twice as fast as using one Jacobi iteration as a preconditioner, at the expense of loosing some parallelism in the sweeps. Other standard preconditioners were studied in this context, such as symmetric successive over-relaxation (SSOR) (see section 10.2 in [117]), but they failed to yield faster convergence, while being more computationally expensive to apply.

Fig. 3-4 depicts the eigenvalues for $\underline{\underline{\mathbf{M}}}$ preconditioned with one block Jacobi iteration and one block Gauss-Seidel iteration. We can observe that for the Gauss-Seidel iteration the eigenvalues are more clustered and there exist fewer outliers. This would explain the fewer number of iteration needed to convergence.

Given the better performance of the Gauss-Seidel iteration we used it as a preconditioner with only one iteration, leading to the preconditioner

$$P^{\text{GS}} \begin{pmatrix} \underline{\mathbf{v}}^{\downarrow} \\ \underline{\mathbf{v}}^{\uparrow} \end{pmatrix} = \begin{pmatrix} (\underline{\mathbf{D}}^{\downarrow})^{-1}\underline{\mathbf{v}}^{\downarrow} \\ (\underline{\mathbf{D}}^{\uparrow})^{-1}\left(\mathbf{v}^{\uparrow} - \underline{\mathbf{L}}(\underline{\mathbf{D}}^{\downarrow})^{-1}\underline{\mathbf{v}}^{\downarrow}\right) \end{pmatrix}. \tag{3.10}$$

The system in Eq. 3.8 is solved using GMRES preconditioned with P^{GS}. Moreover, as in Chapter 2 one can use an adaptive $\mathcal{H}$-matrix fast algorithm for the application of integral kernels, in expressions such as the one above.

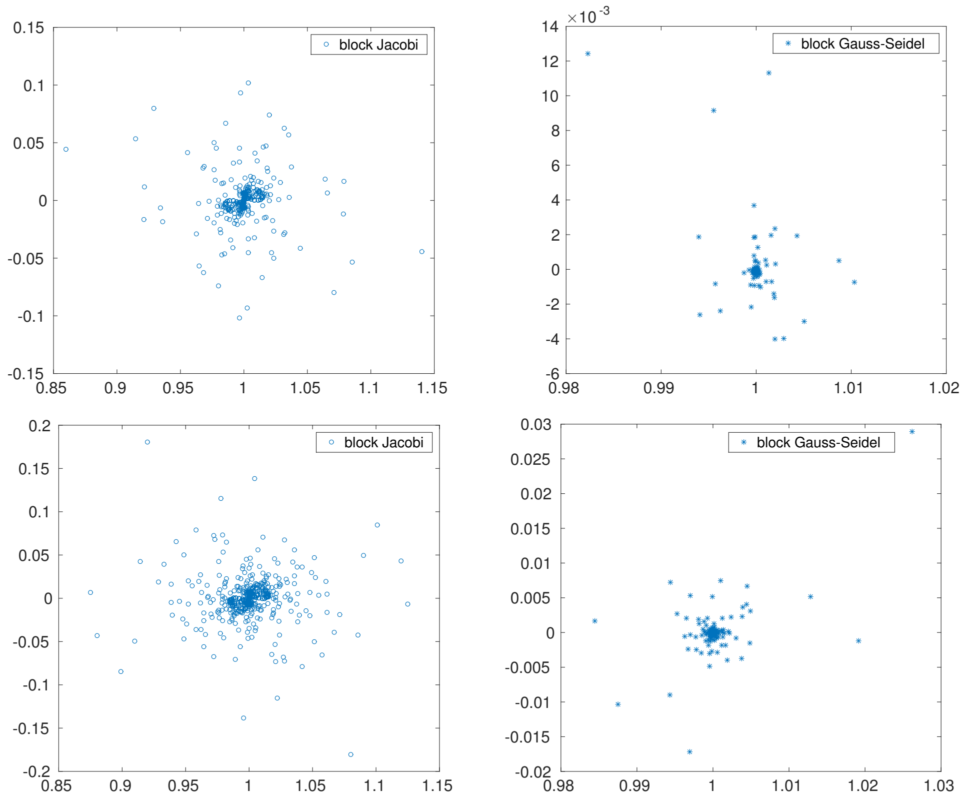


Figure 3-4: Eigenvalues for the preconditioned polarized systems using the block Jacobi (left) and the block Gauss-Seidel (right) preconditioner, using the Marmousi model with $L = 5$, npml $= 10$, and $\omega = 34\pi$ (top row) and $\omega = 70\pi$ (bottom row).

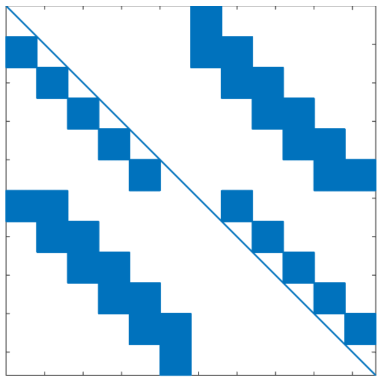


Figure 3-5: Sparsity pattern of the polarized matrix in Eq. 3.9.

Remark 4. *Although the block Gauss-Seidel gives faster convergence; we loose some parallelism. In the case of the block Jacobi iteration, we can apply the sweeps in parallel ($(\underline{\mathbf{D}}^{\uparrow})^{-1}$ and $(\underline{\mathbf{D}}^{\downarrow})^{-1}$), in the block Gauss-Seidel iteration we need to perform then sequentially. In terms of number of floating point operations, the Gauss-Seidel iteration is twice as cheap as the Jacobi iteration; however, in terms of runtime is not clear which will be faster, and it will depend heavily on the implementation.*

Remark 5. *From numerical experiments the convergence rate for the Gauss-Seidel preconditioner seems to have a weaker dependence on the frequency than the Jacobi preconditioner in Chapter 2.*

3.3.2 Matrix-free approach

We proceed to explain the application of $\underline{\underline{\mathbf{M}}}$ and the preconditioner P^{GS} in a matrix-free fashion. The application relies on solving a local problem at each layer with a modified right-hand side. We start with a high-level explanation, then we present the algorithms to apply the preconditioner, and finally, we summarize the matrix-free solver in the form of an algorithm.

One key observation is that the polarized matrix $\underline{\underline{\mathbf{M}}}$ (and $\underline{\mathbf{M}}$) can be applied to a vector in a matrix-free fashion, addressing the offline bottleneck and memory footprint. 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 sources at the boundaries. The application of the Green's integral to a vector $\underline{\mathbf{v}}$, in matrix-free approach, consists in 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 $\underline{\mathbf{M}}$ in a matrix-free fashion is provided in Alg. 9. We use the same notation as in Chapter 2, namely we write

$$\underline{\mathbf{u}} = \left(\mathbf{u}^1_{n^1}, \mathbf{u}^2_1, \mathbf{u}^2_{n^2}, ..., \mathbf{u}^{L-1}_{n^{L-1}}, \mathbf{u}^L_1\right)^t, \tag{3.11}$$

$$\underline{\mathbf{v}} = \left(\mathbf{u}^1_{n^1}, \mathbf{u}^2_1, \mathbf{u}^2_{n^2}, ..., \mathbf{u}^{L-1}_{n^{L-1}}, \mathbf{u}^L_1\right)^t. \tag{3.12}$$

Algorithm 9. *Application of the boundary integral matrix* $\underline{\mathbf{M}}$

```
 1: function  $\underline{\mathbf{u}}$ = BOUNDARY INTEGRAL( $\underline{\mathbf{v}}$ )
 2:     $\widetilde{\mathbf{f}}^1 = -\delta(z_{n^1+1} - z)\mathbf{v}^1_{n^\ell} + \delta(z_{n^1} - z)\mathbf{v}^2_{n^1}$
 3:     $\mathbf{w}^1 = (\mathbf{H}^1)^{-1}\widetilde{\mathbf{f}}^1$
 4:     $\mathbf{u}^\ell_{n^\ell} = \mathbf{w}^\ell_{n^\ell} - \mathbf{v}^\ell_{n^\ell}$
 5:     for $\ell = 2 : L-1$ do
 6:         $\widetilde{\mathbf{f}}^\ell = \delta(z_1 - z)\mathbf{v}^{\ell-1}_{n^{\ell-1}} - \delta(z_0 - z)\mathbf{v}^\ell_1$
                $-\delta(z_{n^\ell+1} - z)\mathbf{v}^\ell_{n^\ell} + \delta(z_{n^\ell} - z)\mathbf{v}^{\ell+1}_{n^{\ell+1}}$
 7:         $\mathbf{w}^\ell = (\mathbf{H}^\ell)^{-1}\widetilde{\mathbf{f}}^\ell$                ▷ inner solve
 8:         $\mathbf{u}^\ell_1 = \mathbf{w}^\ell_1 - \mathbf{v}^\ell_1$
 9:         $\mathbf{u}^\ell_{n^\ell} = \mathbf{w}^\ell_{n^\ell} - \mathbf{v}^\ell_{n^\ell}$
10:     end for
11:     $\widetilde{\mathbf{f}}^L = \delta(z_1 - z)\mathbf{u}^{L-1}_{n^{L-1}} - \delta(z_0 - z)\mathbf{u}^L_1$
```

```
12:     $\mathbf{w}^L = (\mathbf{H}^L)^{-1}\widetilde{\mathbf{f}}^L$
13:     $\mathbf{u}_1^L = \mathbf{w}_1^L - \mathbf{v}_1^L$
14: end function
```

Alg. 9 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 the Gauss-Seidel preconditioner in Eq. 3.10 using the matrix free version. We write

$$\underline{\mathbf{u}}^{\downarrow} = \left(\mathbf{u}_{n^1}^{\downarrow,1}, \mathbf{u}_{n^1+1}^{\downarrow,1}, \mathbf{u}_{n^2}^{\downarrow,2}, ..., \mathbf{u}_{n^{L-1}}^{\downarrow,L-1}, \mathbf{u}_{n^{L-1}+1}^{\downarrow,L-1}\right)^t, \tag{3.13}$$

$$\underline{\mathbf{u}}^{\uparrow} = \left(\mathbf{u}_0^{\uparrow,2}, \mathbf{u}_1^{\uparrow,2}, \mathbf{u}_0^{\uparrow,3}, ..., \mathbf{u}_0^{\uparrow,L}, \mathbf{u}_1^{\uparrow,L}\right)^t, \tag{3.14}$$

to define the components of the polarized wavefields.

The indexes and the arrows are chosen such that they reflect the propagation direction with respect to the superscript. For example, $\mathbf{u}_{n^1}^{\downarrow,\ell}$ represents the wavefield leaving the layer ℓ at the bottom of the layer, i.e propagating downwards and sampled at the bottom of the layer.

We use the notation introduced above to write explicitly the matrix-free operations for the block Gauss-Seidel preconditioner in Eq. 3.10. $\left(\underline{\mathbf{D}}^{\downarrow}\right)^{-1}$ is implemented in Alg. 10, $\left(\underline{\mathbf{D}}^{\uparrow}\right)^{-1}$ is implemented in Alg. 11, and $\underline{\mathbf{L}}$ is implemented in Alg. 12.

Algorithm 10. *Downward sweep*

```
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 11. *Upward sweep*

```
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
```

We observe that in Alg. 10 and 11, the data dependency in the for loop forces the algorithm to be run sequentially. The most expensive operation is the inner solve performed locally at each layer. We will argue in the next section 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.

Algorithm 12. *Upward Reflections*

```
 1: function $\underline{\mathbf{u}}^\uparrow$ = UPWARD REFLECTIONS( $\underline{\mathbf{v}}^\uparrow$ )
 2:     for $\ell = 2 : L-1$ do
 3:         $\mathbf{f}^\ell = \delta(z_1 - z)\mathbf{u}_0^{\uparrow,\ell} - \delta(z_0 - z)\mathbf{u}_1^{\uparrow,\ell}$
                  $-\delta(z_{n^\ell+1} - z)\mathbf{u}_1^{\uparrow,\ell+1} + \delta(z_{n^\ell} - z)\mathbf{u}_0^{\uparrow,\ell+1}$
 4:         $\mathbf{w}^\ell = (\mathbf{H}^\ell)^{-1}\mathbf{f}^\ell$        ▷ inner solve
 5:         $\mathbf{u}_1^{\uparrow,\ell} = \mathbf{w}_1^\ell - \mathbf{v}_1^{\uparrow,\ell}$        ▷ sample the wavefield and subtract the identity
 6:         $\mathbf{u}_0^{\uparrow,\ell} = \mathbf{w}_0^\ell$        ▷ sample the wavefield
 7:     end for
 8:     $\mathbf{f}^L = \delta(z_1 - z)\mathbf{u}_0^{\uparrow,L} - \delta(z_0 - z)\mathbf{u}_1^{\uparrow,L}$
 9:     $\mathbf{w}^L = (\mathbf{H}^L)^{-1}\mathbf{f}^L$        ▷ local solve
10:     $\mathbf{u}_1^{\uparrow,L} = \mathbf{w}_1^L - \mathbf{v}_1^{\uparrow,L}$        ▷ 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. 12 is completely parallel.

The preconditioner in Eq. 3.10 is implemented using these functions, in which the whole block inside the for loop is performed using the reduction depicted in Eq. 3.16, which is detailed in Section 3.3.3.

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 Chapter 2 is the we use Algs. 11, 10, 12 and 9 to perform the GMRES iteration (line 11 of Alg. 13) instead of compressed matrix-vector multiplications.

Algorithm 13. *Matrix-free solver*

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

3.3.3 Nested inner and outer solver

In the presentation of the matrix free solver (Alg. 13), 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. Moreover, as seen before, the nested solver in composed of two levels:

- an inner solve, in which a local system is solved at each layer,
- and an outer solve, in which the global system in the whole domain is solved using the method of polarized traces.

In this section we describe the algorithms to compute the solutions to the inner systems efficiently, and then we describe how the outer solve 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, but this approach requires precomputation and storage of the Green's functions. The matrix-free approach in Alg. 13 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 (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. 10 and 11 show. This means that the careless application of the preconditioner using the matrix-free approach would result in an algorithm with linear online complexity. The nested strategy (that we present below) mitigates this effect 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.

We follow the matrix-free approach of Alg. 13, but instead of a direct solve to invert $\mathbf{H}^\ell$ we use a nested solver, i.e., we use the same reduction used in the whole Ω to each layer Ω^ℓ. We reduce the local problem at each layer to solving a discrete integral system analog to Eq. 3.45 with a layered decomposition in the transverse direction given by

$$\underline{\mathbf{M}}^\ell \underline{\mathbf{u}}^\ell = \underline{\mathbf{f}}^\ell, \qquad \text{for } \ell = 1, .., L_c; \tag{3.15}$$

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. This efficiency can be improved when the inner solver is used in the applications of the Green's integral within the preconditioner. In that case, the application of the Green's integral can be decomposed in 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. 3-6 left); this operation can be represented by a sparse block matrix $\underline{\mathbf{M}}_f^\ell$;

- we solve Eq 3.15 to obtain $\underline{\mathbf{u}}^\ell$ (from pink to blue in Fig. 3-6 right);

- finally, we use the Green's representation formula to sample the wavefield at the interfaces (from blue to green in Fig. 3-6), 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 described above leads to the factorization

$$\begin{bmatrix} \mathcal{G}_0^{\downarrow,\ell}(\mathbf{v}_0, \mathbf{v}_1) + \mathcal{G}_0^{\uparrow,\ell}(\mathbf{v}_{n^\ell}, \mathbf{v}_{n^\ell+1}) \\ \mathcal{G}_1^{\downarrow,\ell}(\mathbf{v}_0, \mathbf{v}_1) + \mathcal{G}_0^{\uparrow,\ell}(\mathbf{v}_{n^\ell}, \mathbf{v}_{n^\ell+1}) \\ \mathcal{G}_{n^\ell}^{\downarrow,\ell}(\mathbf{v}_0, \mathbf{v}_1) + \mathcal{G}_{n^\ell}^{\uparrow,\ell}(\mathbf{v}_{n^\ell}, \mathbf{v}_{n^\ell+1}) \\ \mathcal{G}_{n^\ell+1}^{\downarrow,\ell}(\mathbf{v}_0, \mathbf{v}_1) + \mathcal{G}_{n^\ell+1}^{\uparrow,\ell}(\mathbf{v}_{n^\ell}, \mathbf{v}_{n^\ell+1}) \end{bmatrix} = \underline{\mathbf{M}}_f^\ell \left(\underline{\mathbf{M}}^\ell\right)^{-1} \underline{\mathbf{M}}_u^\ell \cdot \begin{bmatrix} \mathbf{v}_0 \\ \mathbf{v}_1 \\ \mathbf{v}_{n^\ell} \\ \mathbf{v}_{n^\ell+1} \end{bmatrix}, \tag{3.16}$$

in which the blocks of $\underline{\mathbf{M}}_f^\ell$ and $\underline{\mathbf{M}}_u^\ell$ are dense, but highly compressible in PLR form.

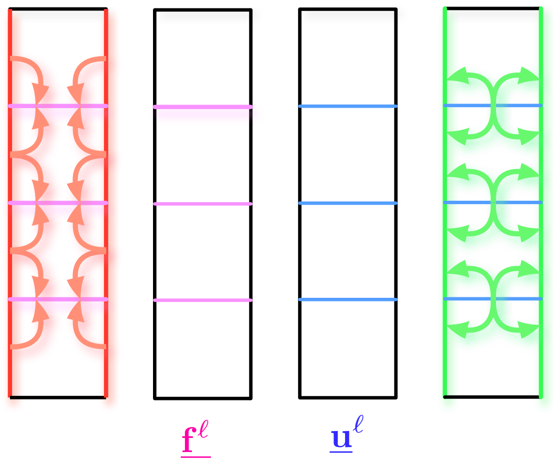


Figure 3-6: Sketch of the application of the Green's functions using a nested approach. The sources are in red (left) and the sampled field in green (right). The application uses the inner boundaries as proxies to perform the solve.

Algorithms

We provide the algorithms in pseudo-code for the two different levels of the nested solver, the inner and outer solvers, Algs. 14 and 15 respectively. In addition, we provide a variant of the inner solver that is crucial to obtain the advertised online complexity at the beginning of the Chapter (i.e. $\mathcal{O}(N/P)$ provided that $P = \mathcal{O}(N^{1/5})$).

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. 14 without modifications. 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. 14. In this variant, the local solves in line 7 of Alg. 14 (which is performed via a LU back-substitution) and the reconstruction (lines 11 to 15 in Alg. 14), are replaced by precomputed operators that are explained in the sequel.

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 and c for the cell 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 $\mathcal{R}^2 = I$ (idempotent). Under the variable swap, we can decompose Ω^ℓ in L_c layers $\{\Omega^{\ell,c}\}_{c=1}^{L_c}$ 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 Chapter 2. Numerically, the variable swap just introduced is implemented by transposing the matrices that represent the different wavefields.

A high-level description of the algorithm is given by Alg. 14

Algorithm 14. *Inner Solve for inverting $\mathbf{H}^\ell$ in Algs. 10, 11 and 12*

```
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:     $\underline{\mathbf{g}}^\ell = \left(\mathcal{N}^{\ell,1}_{n^1}\mathbf{g}^{\ell,1}, \mathcal{N}^{\ell,2}_{1}\mathbf{g}^{\ell,2}, \mathcal{N}^{\ell,2}_{n^2}\mathbf{g}^{\ell,2}, \dots, \mathcal{N}^{\ell,L_c}_{1}\mathbf{g}^{\ell,L_c}\right)^t$          ▷ form r.h.s.
10:     $\underline{\mathbf{v}}^\ell = \left(\underline{\mathbf{M}}^\ell\right)^{-1}\underline{\mathbf{g}}^\ell$                      ▷ solve for the traces (Eq. 3.15)
11:     for $c = 1 : L_c$ do
12:         $\mathbf{v}^{\ell,c}_j = \mathcal{G}^{\uparrow,\ell,c}_j(\mathbf{v}^{\ell,c}_{n^\ell}, \mathbf{v}^{\ell,c}_{n^\ell+1}) + \mathcal{G}^{\downarrow,\ell,c}_j(\mathbf{v}^{\ell,c}_0, \mathbf{v}^{\ell,c}_1) + \mathcal{N}^{\ell,c}_j\mathbf{g}^{\ell,c}$   ▷ local reconstruction
13:     end for
14:     $\mathbf{v}^\ell = (\mathbf{v}^{\ell,1}, \mathbf{v}^{\ell,2}, \dots, \mathbf{v}^{\ell,L_c-1}, \mathbf{v}^{\ell,L_c})^t$          ▷ concatenate the local solutions
15:     $\mathbf{w} = \mathcal{R} \circ \mathbf{v}^\ell$                                     ▷ variable swap
16: end function
```

Using the inner solve defined in Alg. 14 we can provide a high level description of the online stage of the nested solver in Alg. 15.

Algorithm 15. *Outer solver, or online computation for the nested solver*

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

```
22: end function
```

For the sake of clarity, we decompose the outer solve in Alg. 15 in three stages :

- *lines 2-10:* preparation of the r.h.s. for the outer polarized integral system, using an inner solve in each layer concurrently;
- *lines 11-12:* solving 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;
- *lines 13-21:* reconstruction of the solution inside the volume at each node, using an inner solve in each layer concurrently.

The computational cost incurred using the implementation of Alg. 14 is acceptable if $\mathbf{f}^{\ell}$ has no *a priori* structure, or if the wavefield is needed the in volume. This is the case in the preparation of the r.h.s. and the reconstruction of the solution. However, within the GMRES loop that solves for the traces using the matrix-free approach (line 11 in Alg. 15), we have that the support of the source term is concentrated on the boundaries between layers (see line 5 in Algs. 11, 10; lines $3, 6$ and 12 in Alg. 9 and lines 3 and 8 in Alg. 12).

When the inner solve is used in the preconditioner (when applying the Green's integrals), we use the localization of the support to replace the solve inside each cell by the direct application of the Green's function to the source term. We use the Green's functions $G^{\ell,c}(\mathbf{x},\mathbf{y})$, where $\mathbf{y}$ lies on the support of the sources and $\mathbf{x}$ on the boundary between cells, to build $\underline{\mathbf{g}}^{\ell}$ (line 9 in Alg. 14). We precompute the matrix that encodes this operation; the resulting matrix can be easily compressed in PLR form to obtain a fast matrix-vector product.

Simultaneously, we can observe that the output of Algs. 11, 10 and 12 consists of the traces of the solution at the interfaces between layers; all the degrees of freedom at the interior are unnecessary. We use this fact to further reduce the amount of operations to the strictly necessary. We only sample the Green's representation formula at the boundaries between layers when reconstructing the solution (line 12 of Alg. 14). The matrix that encodes this operation can be precomputed, and once again the resulting matrices are highly compressible in PLR form.

The matrices $\underline{\mathbf{M}}_f^{\ell}$ and $\underline{\mathbf{M}}_u^{\ell}$ are block matrices whose blocks are the matrices described above.

The choice of algorithm to solve Eq. 3.15 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.

Nested polarized traces

To efficiently apply the Green's integrals using Alg. 14, we need to solve Eq. 3.15 efficiently. One alternative is to use the method of polarized traces to solve the system at each layer, this approach will be called the method of nested polarized traces. Following Chapter 2 this approach has the same empirical scalings, at the inner level, as those found it [148] when the blocks are compressed in (PLR) form. Each layer solve has $\mathcal{O}(L_c(n/L + \log(n))^{3/2})$ online complexity when $\omega \sim \sqrt{n}$.

Given that $L_c \sim L \sim \sqrt{P}$, we have that each solve is done in $\mathcal{O}(N^{3/4}/L + L\log(n)^{3/2})$ time, which has an extra factor $1/L$ when compared with the direct compressed matrix-vector multiplication for the application of the Green's integral in Chapter 2. Although 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.

Inner compressed-block LU

An alternative to efficiently apply the Green's integrals via Alg. 14, is to use the compressed-block LU to solve Eq. 3.15. Given the banded structure of $\underline{\mathbf{M}}^\ell$ (see Fig. 3-5), we perform a block LU decomposition without pivoting. The resulting LU factors are block sparse and tightly banded (see Fig. 3-2). We have the factorization

$$\underline{\mathbf{M}}^\ell = \underline{\mathbf{L}}^\ell \, \underline{\mathbf{U}}^\ell, \tag{3.17}$$

which leads to

$$\mathcal{G}^\ell = \underline{\mathbf{M}}^\ell_f \left(\underline{\mathbf{U}}^\ell\right)^{-1} \left(\underline{\mathbf{L}}^\ell\right)^{-1} \underline{\mathbf{M}}^\ell_u, \tag{3.18}$$

in which $\mathcal{G}^\ell$ represents the linear operator at the left-hand-side of Eq.3.16. Following Section 3.2.2 we have that solving $\left(\underline{\mathbf{U}}^\ell\right)^{-1} \left(\underline{\mathbf{L}}^\ell\right)^{-1}$ can be done in $\mathcal{O}(L_c(n/L + \log(n))^{3/2})$ online time. The main advantage 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 as a layer solve.

3.3.4 Complexity

Table 3.3 summarizes the complexities and number of processors at each stage for both methods. 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.

If the frequency scales as $\omega \sim \sqrt{n}$, the regime in which second order finite-differences are expected to be accurate, we obtain $\alpha = 5/8$; however, we assume the more conservative value $\alpha = 3/4$. The latter is in better agreement with a theoretical analysis of the rank of the off-diagonal blocks of the Green's functions. In such scenario we have that the blocks of $\underline{\mathbf{M}}^\ell_u$ and $\underline{\mathbf{M}}^\ell_f$ can be compressed in PLR form, resulting in a fast application in $\mathcal{O}(L_c(n/L + \log(n))^{3/2})$ time, easily parallelizable among L_c nodes. Solving Eq. 3.15 can be solved using either the direct compressed or the nested polarized traces in $\mathcal{O}(L_c(n/L + \log(n))^{3/2})$ time. This yields a runtime of $\mathcal{O}(L_c(n/L + \log(n))^{3/2})$ for each application of the Green's integral.

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

Table 3.3: Complexity of the different steps of the preconditioner, 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. Typically $\alpha = 3/4$.

N	$\omega/2\pi[Hz]$	10×2	40×8	100×20
88×425	7.71	**(3)** 0.89	**(3)** 15.6	**(4)** 97.9
175×850	11.1	**(3)** 1.48	**(3)** 17.7	**(3)** 105
350×1700	15.9	**(3)** 2.90	**(3)** 22.1	**(4)** 106
700×3400	22.3	**(3)** 5.58	**(3)** 31.3	**(4)** 126
1400×6800	31.7	**(3)** 10.5	**(3)** 47.9	**(4)** 176

Table 3.4: 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 Marmousi2 model (see [95]).

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 3.5: 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 [19]).

To apply the Gauss-Seidel preconditioner we need $\mathcal{O}(L)$ applications of the Green's integral, resulting in a runtime of $\mathcal{O}(L\cdot L_c(n/L+\log(n))^{3/2})$ to solve Eq. 3.8. Using the fact that $L\sim\sqrt{P}$ and $L_c\sim\sqrt{P}$ and adding the contribution of the other steps of the online stage; we have that the overall online runtime is given by $\mathcal{O}(P^{1/4}N^{3/4}+$

$P\log(N)^{3/4}+N/P+\log(N)^2)$, which is $\mathcal{O}(N/P)$ (up to logarithmic factors) provided that $P \lesssim N^{1/5}$.

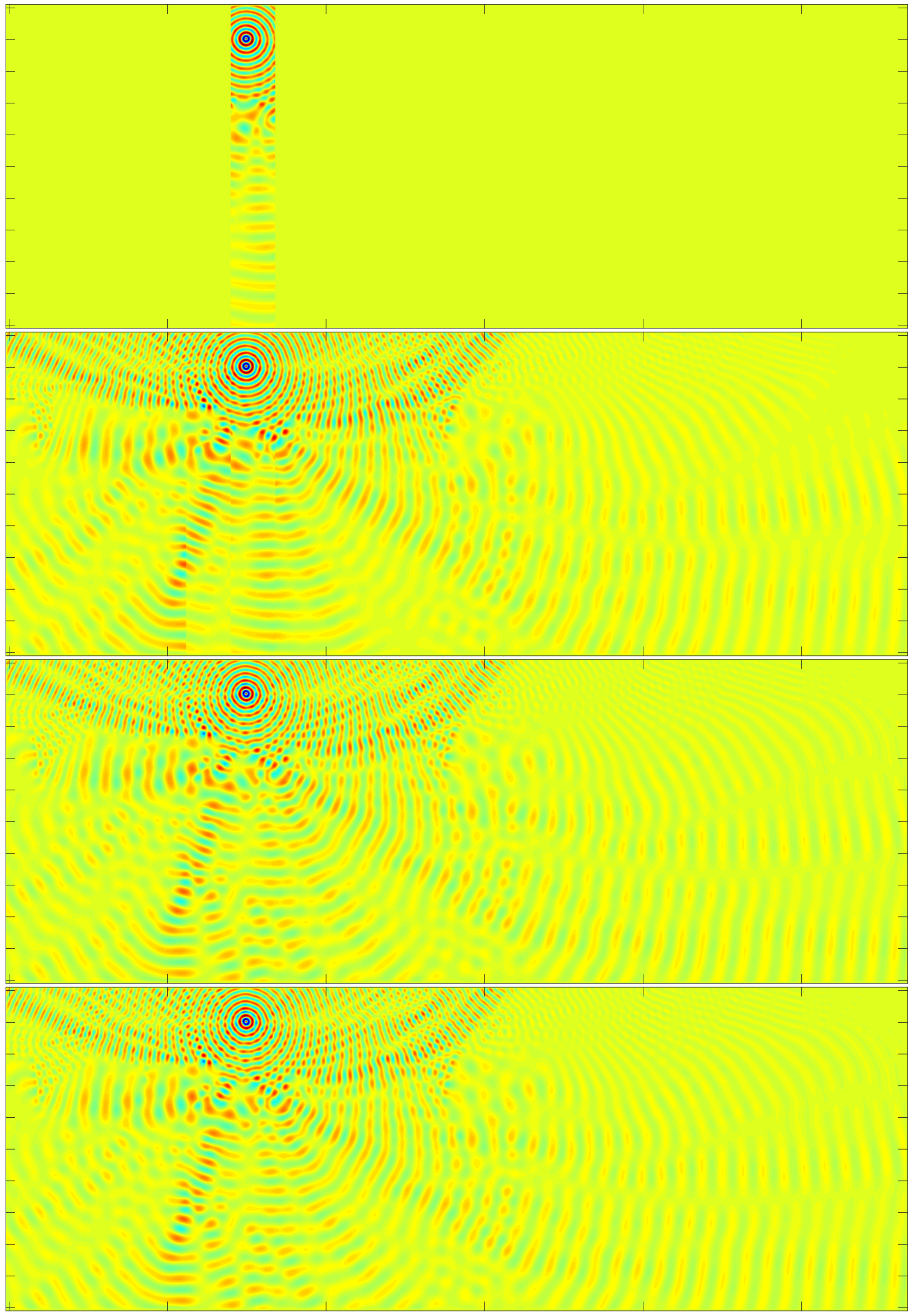


Figure 3-7: Two iteration of the preconditioner, from top to bottom: initial guess with only local solve; first iteration, second iteration, final solution. The background model is given by the BP 2004 model [19].

Finally, the memory footprint is $\mathcal{O}(P^{1/4}N^{3/4} + P\log(N)^{3/4} + N/P + \log(N)^2)$ and the communication cost for the online part is $\mathcal{O}(n\sqrt{P})$, which represents an asymptotic improvement with respect to [148], in which the storage and communication cost are $\mathcal{O}(PN^{3/4} + N/P + \log(N)^2)$ and $\mathcal{O}(nP)$, respectively.

3.4 Numerical results

3.4.1 Compressed-block LU

We present some numerical examples for the inner boundary reduction for media featuring large resonant cavities with sharp contrasts, and we provide some numerical examples to illustrate the suboptimal behavior of the Chapter 2's polarized traces formulation, as well as the nested polarized traces formulation in this Chapter, for this particular problem.

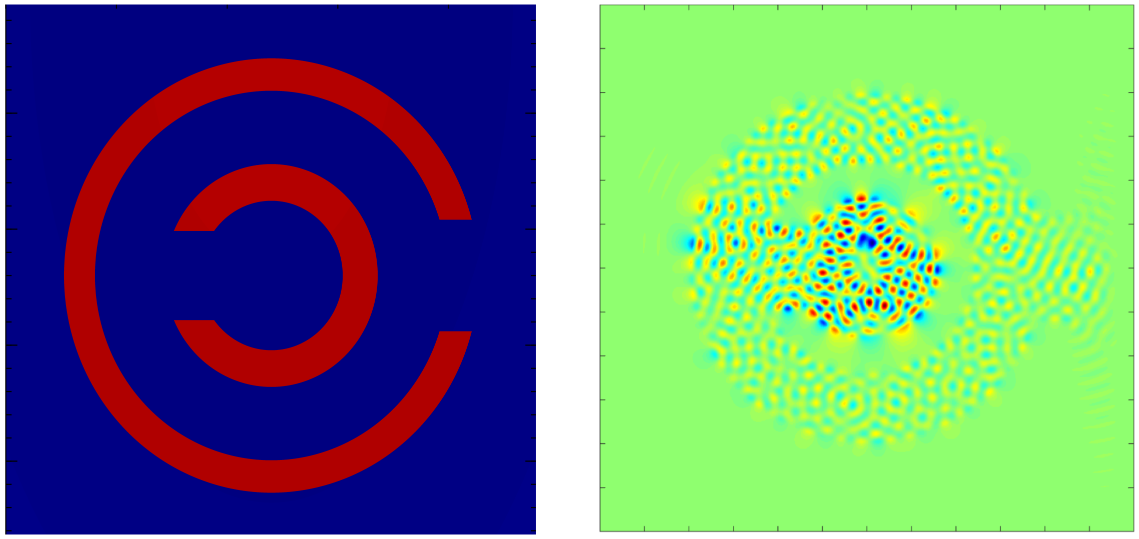


Figure 3-8: Left: Resonant wave-guide inspired in the Comedy Central logo; right: typical solution.

The model in Fig. 3-8 (left) is used for the numerical experiments. The shape is kept constant, with an adjustable c_{red} and with a smooth background speed $c_{\text{blue}}(x, y) = 1 + 0.1x + 0.1y$ for $(x, y) \in [0, 1]^2$, such that the problem can not be reduced to an integral equations posed on the boundaries. The model in Fig. 3-8 (left) was provided as a function handle, to allow an arbitrarily fine sampling close to its sharp interfaces, which are not aligned with a regular equispaced grid. Discretizing the Helmholtz equation with finite difference would incur in a severe reduction of the accuracy. In practice, a good accuracy is crucial to obtain the reported compression scalings for the Green's integrals, given that we compress a numerical approximation of the Green's functions. In general, an incorrect discretization will deteriorate greatly the compression of the Green's integrals in PLR form in the high-frequency regime, for a fixed compression accuracy ϵ.

To improve the accuracy, we use a Q1 discretization of Eq. 2.3, with a symmetric formulation and an adaptive quadrature at the sharp interfaces as explained in Appendix 3.A. This different discretization leads to a different Green's representation formula, which is derived in Appendix 3.B. Fortunately, all the machinery developed in Chapter 2 extends naturally to Q1 finite elements.

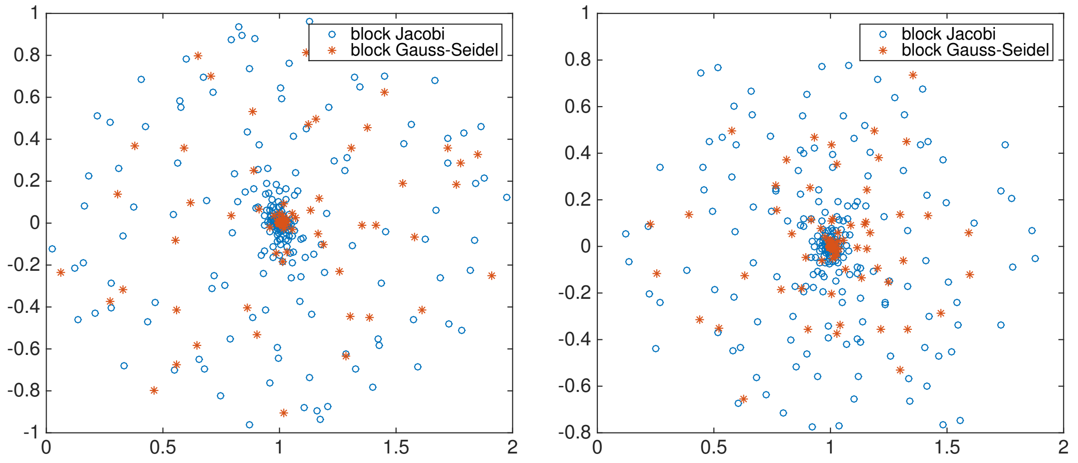


Figure 3-9: Eigenvalues in the complex plane for the preconditioned polarized system using the model in Fig. 3-8 for different contrasts left: $c_{\text{red}} = 10$; right: $c_{\text{red}} = 2$.

Fig 3-9 shows the eigenvalues of the preconditioned polarized system, using $L = 4$ layers, $\omega = 40\pi$ for different contrasts. We can observe that the eigenvalues are spread in the unit disk centered at one. Moreover, as the contrast is increased, more eigenvalues live near the circle of radius 1 centered at one, and we can observe some eigenvalues close to zero. In such circumstances, GMRES is known to have a hard time converging to the solution. We point out the radically different behavior from the spectra shown in Fig. 3-4, in which all the eigenvalues are tightly clustered around 1, far from zero.

The results for $L = 10$ layers, different contrasts, frequencies, and problem sizes, are shown in Fig.3-10. We observe that the runtimes are independent of the contrast at high-frequency. The complexity of the method of polarized traces would severely deteriorate with high contrasts as Fig 3-9 shows, given the large amount of iterations needed for convergence.

Fig. 3-11 shows the runtime for a fixed constrast ($c_{\text{red}} = 100$) and shows the scaling for the fast solve of Eq. 3.5. We obtain the same scaling as in [148] for the cavity-free problem.

To compare against other methods, we apply the same compressed block LU technique to solve a Schur complement system associated to a layered partitioning, which is explained Appendix 3.C, rather than the GRF-based Eq. 3.5, we empirically obtain the same asymptotic scalings, albeit with smaller constants given that the Schur complement system is roughly half the size of the boundary integral system given by the GRF.

As mentioned before, in principle, the cost of the of offline computation of the boundary-reduced LU factors and their compression can be decreased by using $\mathcal{H}$-matrix algebra to perform the elimination.

Remark 6. *Each local problem for the Schur complement is, generally speaking, a waveguide in the longitudinal direction. This results on the blocks of* $\underline{\mathbf{S}}$ *to have off diagonal blocks of higher rank than the ones in* $\underline{\mathbf{M}}$ *as noted in [61]. However, once the LU factors are computed and compressed, both methods have comparable asymptotic runtimes.*

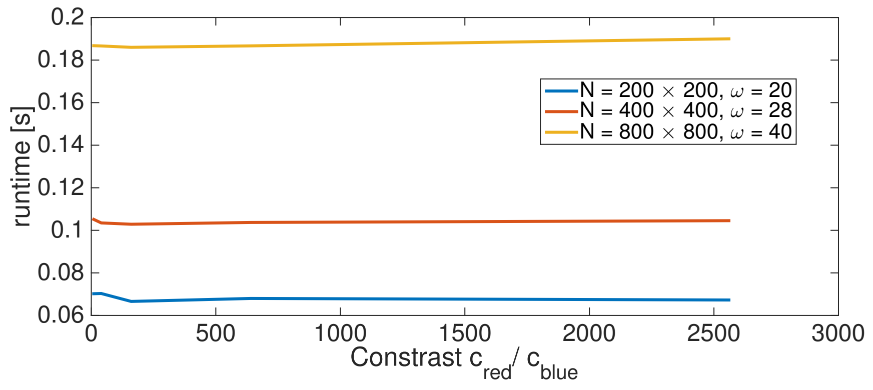


Figure 3-10: Online runtime for different constrasts and problem sizes. L is fixed throughout.

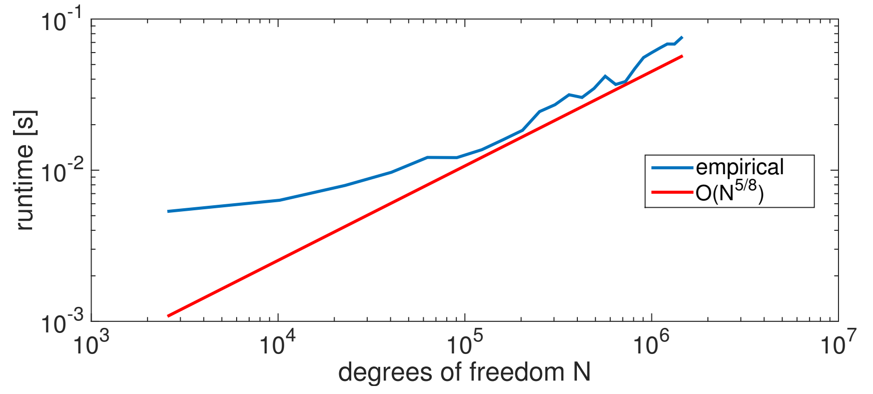


Figure 3-11: Runtime for a fixed constrast. L is fixed throughout.

3.4.2 Nested Solver

Fig. 3-7 depicts the fast convergence of the method. After a couple of iterations the exact and approximated solution are indistinguishable to the naked eye.

Tables 3.4 and 3.5 show the sublinear scaling for one GMRES iteration, with respect to the degrees of freedom in the volume. We can observe that the number of iterations to converge depends weakly on the frequency and the number of subdomains. Fig. 3-12 shows the empirical scaling for one global GMRES iteration, in which the maximum rank ϵ-rank in the adaptive PLR compression scales as $\max_{\text{rank}} \sim \sqrt{\omega}$ and $\epsilon = 10^{-8}$. Moreover, for the nested polarized traces, the accuracy for the GMRES inner solve is fixed to 10^{-6}. We can observe that both methods have the same asymptotic runtime, but with different constants.

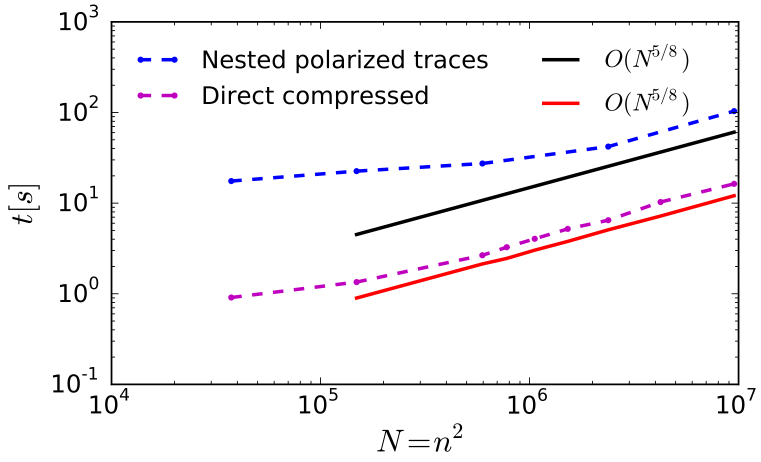


Figure 3-12: Runtime for one GMRES iteration using the two different nested solves, for $L = 9$ and $L_c = 3$, and $\omega \sim \sqrt{n}$.

We point out that some gains can be made by using different compressed operators, one highly accurate to apply $\underline{\underline{\mathbf{M}}}$, an operation that is easily parallelizable, and another with low accuracy to apply the preconditioner, an operation that is completely sequential.

3.5 Conclusion

We presented an extension to the method of polarized traces introduced in Chapter 2, with improved asymptotic runtimes in a distributed memory environment. The method has sublinear runtime even in the presence of rough media of geophysical interests. Moreover, its performance is completely agnostic to the source.

This algorithm is of especial interest in the context of time-lapse full-waveform inversion, continuum reservoir monitoring, and local inversion. If the update to the model is localized, most of precomputations can be reused. The only extra cost is the refactorization and computation of the Green's functions in the cells with a non null intersection with the support of the update, reducing sharply the computational cost.

We point out that this approach can be further parallelized using distributed algebra libraries. Moreover, the sweeps can be pipelined to maintain a constant load among all the nodes.

Appendix

3.A Discretization using Q1 finite elements

We pose Eq. 3.1 with absorbing boundary conditions on $\partial\Omega$, realized as a perfectly matched layer (PML) [13, 85].

Let $\Omega^{\text{ext}} = (-\delta_{\text{pml}}, L_x + \delta_{\text{pml}}) \times (-\delta_{\text{pml}}, L_z + \delta_{\text{pml}})$ be the extended rectangular domain containing Ω and its absorbing layer. The symmetric formulation of the Helmholtz equation takes the form

$$-\left(\nabla \cdot \Lambda \nabla + \frac{\omega^2 m(\mathbf{x})}{\alpha_x(\mathbf{x})\alpha_z(\mathbf{x})}\right) u(\mathbf{x}) = \frac{f(\mathbf{x})}{\alpha_x(\mathbf{x})\alpha_z(\mathbf{x})}, \tag{3.19}$$

where

$$\Lambda(\mathbf{x}) = \left[\begin{array}{cc} s_x(\mathbf{x}) & 0 \\ 0 & s_z(\mathbf{x}) \end{array}\right], \tag{3.20}$$

$s_x = \alpha_x/\alpha_z$, $s_z = \alpha_z/\alpha_x$, where α_x and α_z are defined in Eq. 2.13 and similarly for $\sigma_z(\mathbf{x})$.

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, highly sophisticated quadratures and adaptive meshes have to be implemented to properly approximate the finite difference operator [145, 4]. We opted for a low order Q1 finite element discretization, with an adaptive quadrature rule at the discontinuities.

Eq. 3.19 is discretized using Q1 elements leading to a discretized matrix

$$\mathbf{H} = \mathbf{S} - \mathbf{M}, \tag{3.21}$$

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; and adaptive trapezoidal rule is used, until a preset accuracy is achieved.

To discriminate if 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 is is considered discontinuous.

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

$$\mathbf{H}\mathbf{u} = \mathbf{f}, \tag{3.22}$$

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 gauss quadrature rule.

The discretization is second order accurate even in the case of discontinuous interfaces with sharp contrasts, as long as the adaptive quadrature rule is used to ensure a small error on the numerical integration.

Finally, we can apply the same Q1 discretization to every local problem given by Eq. 3.2 obtaining

$$\mathbf{H}^{\ell}\mathbf{u}^{\ell} = \mathbf{f}^{\ell}. \tag{3.23}$$

We point out that for the local problems we do not use the normal extension of the slowness in the PML, instead we use the neighboring values on the wave speed in the damping layer.

In the sequel, we will perform extensive manipulations on the matrix $\mathbf{H}$. In order to minimize the burden of the notations, we assume the same ordering as in Appendix 2.A, i.e.

$$\mathbf{u} = (u_{1,1}, u_{2,1}, ..., u_{n_x,1}, u_{1,2}, ..., u_{n_x,2}, ..., u_{n_x,n_z}), \tag{3.24}$$

and we use the notation

$$\mathbf{u}_n = (u_{1,n}, u_{2,n}, ..., u_{n_x,n}), \tag{3.25}$$

i.e. $\mathbf{u}$ sampled at constant depth.

Following that ordering we write $\mathbf{H}$ as a block matrix in the form

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_{1,1} & \mathbf{H}_{1,2} & & & \\ \mathbf{H}_{2,1} & \mathbf{H}_{2,2} & \mathbf{H}_{2,3} & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \mathbf{H}_{n_z-1,n_z} \\ & & & \mathbf{H}_{n_z,n_z-1} & \mathbf{H}_{n_z,n_z} \end{bmatrix}, \tag{3.26}$$

in which each block correspond to a fixed z.

We can generalized the boundary integral system in Chapter 3.1 using the Green's representation formula in Appendix 3.B resulting in the equivalent problem

$$\underline{\mathbf{M}\mathbf{u}} = \underline{\mathbf{f}}. \tag{3.27}$$

Where

$$\underline{\mathbf{M}} = \frac{1}{h}\begin{bmatrix} -\mathbf{G}^1_{n,n+1} - \mathbf{I} & \mathbf{G}^1_{n,n} & 0 & 0 & 0 & 0 \\ \mathbf{G}^2_{1,1} & -\mathbf{G}^2_{1,0} - \mathbf{I} & -\mathbf{G}^1_{1,n+1} & \mathbf{G}^1_{1,n} & 0 & 0 \\ \mathbf{G}^2_{n,1} & -\mathbf{G}^2_{n,0} & -\mathbf{G}^2_{n,n+1} - \mathbf{I} & \mathbf{G}^2_{n,n} & 0 & 0 \\ 0 & \ddots & \ddots & \ddots & 0 & 0 \\ 0 & 0 & \mathbf{G}^{L-1}_{1,1} & -\mathbf{G}^{L-1}_{1,0} - \mathbf{I} & -\mathbf{G}^{L-1}_{1,n+1} & \mathbf{G}^{L-1}_{1,n} \\ 0 & 0 & \mathbf{G}^{L-1}_{n,1} & -\mathbf{G}^{L-1}_{n,0} & -\mathbf{G}^{L-1}_{n,n+1} - \mathbf{I} & \mathbf{G}^{L-1}_{n,n} \\ 0 & 0 & 0 & 0 & \mathbf{G}^{L}_{1,1} & -\mathbf{G}^{L}_{1,0} - \mathbf{I} \end{bmatrix}, \tag{3.28}$$

and the Green's integral are defined as follows:

$$\mathbf{G}^\ell_{1,0} = -\delta_1\left((\mathbf{H}^\ell)^{-1}\delta_0\mathbf{H}^\ell_{1,0}\right), \tag{3.29}$$

$$\mathbf{G}^\ell_{1,1} = -\delta_1\left((\mathbf{H}^\ell)^{-1}\delta_1\mathbf{H}^\ell_{0,1}\right), \tag{3.30}$$

$$\mathbf{G}^\ell_{n,n} = -\delta_n\left((\mathbf{H}^\ell)^{-1}\delta_{n+1}\mathbf{H}^\ell_{n+1,n}\right), \tag{3.31}$$

$$\mathbf{G}^\ell_{n,n+1} = -\delta_n\left((\mathbf{H})^{-1}\delta_n\mathbf{H}_{n,n+1}\right). \tag{3.32}$$

Moreover,

$$\mathbf{I} = I/h \tag{3.33}$$

is a rescaled identity to reduce the notational burden.

Remark 7. *The Green's functions are still computed by inverting $\mathbf{H}^\ell$ using a Dirac delta as a point source. However, given that we are dealing with finite elements a point source is defined differently. Using the formulas above we recover the same Green's functions in the case of a finite difference discretization.*

3.B Green's representation formula

We present a generalization of the domain decomposition framework developed in Chapter 2 to Q1 regular finite elements.

We start by providing the algebraic formula for the discrete Green's representation formula, and we propose a technique to derive such formulas without the time consuming computations performed in Appendix 2.A. We point out that there are clear parallels between this derivation and the reduction to an interface problem using interior Schur complements.

We want to derive the algebraic formula for the Green's representation formula. From Theorem 1 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.

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

$$\mathbf{v}^\ell = \mathbf{u}\chi_{\Omega^\ell}, \tag{3.34}$$

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

$$\mathbf{H}^\ell \mathbf{v}^\ell = \mathbf{f}^\ell + \mathcal{F}^\ell(\mathbf{u}), \tag{3.35}$$

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 Eq. 3.35 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 3.34.

For ℓ fixed we can obtain the expression of $\mathcal{F}^\ell$ by evaluating Eq. 3.35 and imposing that $\mathbf{v}^\ell = \mathbf{u}\chi_{\Omega^\ell}$. In particular, we need to evaluate Eq. 3.35 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 Eq. 3.35 at $k = 1$, we have that

$$\mathbf{H}^\ell_{1,1}\mathbf{v}^\ell_1 + \mathbf{H}^\ell_{1,2}\mathbf{v}^\ell_2 = \mathbf{f}^\ell_1 + \mathcal{F}_1(\mathbf{u}). \tag{3.36}$$

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

$$\mathbf{H}_{1,0}\mathbf{u}_0 + \mathbf{H}_{1,1}\mathbf{u}_1 + \mathbf{H}_{1,2}\mathbf{u}_2 = \mathbf{f}^\ell_1. \tag{3.37}$$

By imposing that $\mathbf{v}^\ell = \mathbf{u}\chi_{\Omega^\ell}$ and subtracting Eqs 3.36 and 3.37, we have that

$$\mathcal{F}^\ell_1(\mathbf{u}) = -\mathbf{H}_{1,0}\mathbf{u}_0 = -\mathbf{H}^\ell_{1,0}\mathbf{u}_0. \tag{3.38}$$

We can observe that the role of $\mathcal{F}^\ell$ is to complete Eq. 3.35 a 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 Eq. 3.35 can be evaluated at $k = 0$ obtaining

$$\mathbf{H}^\ell_{0,1}\mathbf{v}^\ell_1 = \mathcal{F}^\ell_0(\mathbf{u}), \tag{3.39}$$

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

$$\mathcal{F}^\ell_0(\mathbf{u}) = \mathbf{H}_{0,1}\mathbf{u}_1 = \mathbf{H}^\ell_{0,1}\mathbf{u}_1. \tag{3.40}$$

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

$$\mathcal{F}^\ell_k(\mathbf{u}) = 0. \tag{3.41}$$

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

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

which can be replaced on Eq. 3.35 leading to

$$\begin{aligned}\mathbf{H}^\ell \mathbf{v}^\ell = &- \delta_{n^\ell} \mathbf{H}^\ell_{n^\ell, n^\ell+1} \mathbf{u}_{n^\ell+1} + \delta_{n^\ell+1} \mathbf{H}^\ell_{n^\ell+1, n^\ell} \mathbf{u}_{n^\ell} \\ &- \delta_1 \mathbf{H}^\ell_{1,0} \mathbf{u}_0 + \delta_0 \mathbf{H}^\ell_{0,1} \mathbf{u}_1 + \mathbf{f}^\ell. \end{aligned} \tag{3.42}$$

Eq. 3.42 can be transformed to 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 Eq. 2.24 in the form

$$\mathcal{G}_j^{\downarrow,\ell}(\mathbf{v}_0, \mathbf{v}_1) = h \left[\begin{array}{cc} \mathbf{G}^\ell(z_j, z_1) & \mathbf{G}^\ell(z_j, z_0) \end{array} \right] \begin{pmatrix} -\mathbf{H}^\ell_{1,0}\mathbf{v}_0 \\ \mathbf{H}^\ell_{0,1}\mathbf{v}_1 \end{pmatrix}, \tag{3.43}$$

$$\mathcal{G}_j^{\uparrow,\ell}(\mathbf{v}_{n^\ell}, \mathbf{v}_{n^\ell+1}) = h \left[\begin{array}{cc} \mathbf{G}^\ell(z_j, z_{n^\ell+1}) & \mathbf{G}^\ell(z_j, z_{n^\ell}) \end{array} \right] \begin{pmatrix} \mathbf{H}^\ell_{n^\ell+1,n^\ell}\mathbf{v}_{n^\ell} \\ -\mathbf{H}^\ell_{n^\ell,n^\ell+1}\mathbf{v}_{n^\ell+1} \end{pmatrix}. \tag{3.44}$$

Finally, if we redefine $\mathbf{G}^\ell(z_j, z_k)$ for $k = 0, 1, n^\ell, n^\ell + 1$, so they absorb all the extra factors in Eqs. 3.43 and 3.44, we can use all the machinery introduced in Chapter 2 to define the boundary integral system and the polarized systems.

We build the boundary integral system

$$\underline{\mathbf{M}}\underline{\mathbf{u}} = \underline{\mathbf{f}}, \tag{3.45}$$

although we do not provide a rigorous proof that solving the equation above is equivalent to solving Eq. 3.1, we have exhaustive numerical evidence that the later is true.

Remark 8. *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 2.A.*

3.C Schur Complement

We decompose the system $\mathbf{Hu} = \mathbf{f}$ in L different layers $\{\Omega^\ell\}_{\ell=1}^L$, using $L-1$ separators $\{\Gamma_{\text{Schur}}^{\ell,\ell+1}\}_{\ell=1}^{L-1}$ as depicted in Fig. 3.13.

Using standard linear algebra we reduce the discrete PDE to solving the Schur complements on the separators. Leading to a problem of the form

$$\underline{\mathbf{S}}\underline{\mathbf{u}} = \underline{\mathbf{f}}_{\text{schur}}, \tag{3.46}$$

in which $\underline{\mathbf{S}}$ is a block tridiagonal matrix, with dense blocks; and $\underline{\mathbf{u}}$ are the stacked traces of $\mathbf{u}$ at the separators. Moreover, $\underline{\mathbf{f}}_{\text{schur}}$ is the source, which contains elements from the local solves, to solve the Schur complement system. Once $\underline{\mathbf{u}}$ is computed we solve the local problems with the correct Dirichlet data to recover the solution at each layer.

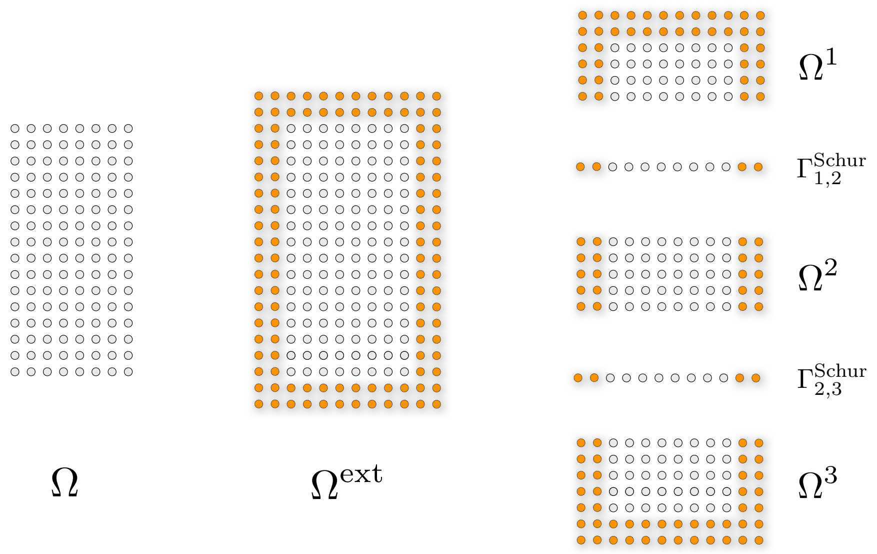


Figure 3.13: Sketch of the domain decomposition for the Schur complement.

We point out that in this case the local problems are essentially different from the ones solved for the Green's representation formula. Instead of solving a local Helmholtz equation with artificial absorbing boundary conditions at the interfaces between layers, we solve a local Hemholtz problem with a Dirichelet boundary condition. The latter problem can be ill-posed if the frequency is such that we are in presence of a resonant mode. Fortunately, we have absorbing boundary conditions in the longitudinal direction, which prevent us to having a local resonant mode as pictured in Fig. 3.13.

Chapter 4

Harmonic Extrapolation

This Chapter is concerned with the problem of extension, or "extrapolation" of harmonic wavefields, i.e., finding solutions of the two-dimensional Helmholtz equation $\triangle u + \kappa^2 u = 0$ in the half-space $(x, y) \in \mathbb{R} \times \mathbb{R}_0^+$ from boundary data at $y = 0$. We study the problem of stability of such extensions in the presence of partial data in a finite interval of $\{y = 0\}$.

This Chapter provides a theoretical rationale to the method of polarized traces, which can be seen as an algebraic extrapolation technique. Within the scope of domain decomposition used in Chapters 2 and 3 the extrapolation can be seen as follows: given a Dirichlet and Neumann trace of the solution at one of the interfaces of a slab, we seek the minimum conditions needed to be able to extrapolate, or to guess the value of the solution at the opposite interface. Algebraically, the elimination of unknowns provides an answer for the discretized system. Indeed, the elimination can be done by running the recurrence relation given by the discrete system as is done in the proofs in Appendix 2.C. However, the elimination of unknowns has to be constructed following the physics of the wave equation that hinges on the propagation of singularities, in particular, on the causality of waves. This physical intuition is exploited in most of the highly efficient Helmholtz preconditioners, such as [58, 59, 143, 37, 128, 148], using absorbing boundary conditions to enforce some kind of local directionality, hence causality to the waves.

In this Chapter we study the analytical properties and limitations of the extrapolation procedure. We provide some asymptotic geometrical conditions necessary for an accurate extrapolation, followed by numerical experiments that corroborate the asymptotics. Finally, we present a sweeping-like solver of a different nature compared to the others solvers in this thesis. The solver uses the extrapolation to compute the solution to the Helmholtz equation in homogeneous medium very accurate in a single sweep.

4.1 Harmonic extension: the Laplace equation

The $\kappa = 0$ case corresponds to the problem of extending an analytic – or classically harmonic – function from its knowledge on a line or curve, a fundamental question linked to the genesis of Fourier analysis in the first half of the 19th century. Siméon Poisson [110] wrote the general solution of the Laplace equation when the domain is either a disk, or a half-space with decay conditions. In the latter case, he found that the solution of

$$\triangle u = 0 \qquad \mathbf{x} = (x,y) \in \mathbb{R} \times \mathbb{R}_0^+, \tag{4.1}$$

$$u(x,0) = u_0(x) \qquad x \in \mathbb{R}, \tag{4.2}$$

$$u = \mathcal{O}\left(\frac{1}{x^2+y^2}\right) \qquad \text{as } x^2+y^2 \to \infty, \tag{4.3}$$

when u_0 decays sufficiently fast, is the convolution integral

$$u(x,y) = \frac{1}{\pi}\int_{\mathbb{R}} P_y(x-x')u_0(x')dx', \tag{4.4}$$

with the (Poisson) kernel

$$P_y(x) = \frac{y}{x^2+y^2}. \tag{4.5}$$

This map from u_0 to u is manifestly stable and smoothing[1] for any $y > 0$, but this behavior is fragile. For instance, the extension problem becomes highly ill-posed for all $y > 0$ when the match with u_0 is only required in any finite segment γ of $\{y = 0\}$, i.e., when the knowledge of u_0 is *partial*.

Indeed, any real-analytic $u_0(x)$ admits real-analytic approximations $\widetilde{u}_0(x)$ which are arbitrarily close on γ, but arbitrarily far outside of it — it suffices for instance to add to u_0 a high-amplitude gaussian centered well away from γ. The Poisson kernel integrates constructively with the perturbation, and the predictions $u(x,y)$ and $\widetilde{u}(x,y)$ can be arbitrarily far from one another even for very small $y > 0$. This behavior is typical: boundary-value elliptic equations cannot in general be solved by marching from incomplete boundary data.

4.2 Generalized harmonic extension: the Helmholtz equation

The situation is different when the frequency parameter κ becomes large. We now consider the Helmholtz equation in the upper half plane with outgoing (Sommerfeld) radiating boundary conditions, namely

$$\triangle u + \kappa^2 u = 0 \qquad \mathbf{x} = (x,y) \in \mathbb{R} \times \mathbb{R}_0^+, \tag{4.6}$$

$$u(x,0) = u_0(x) \qquad x \in \mathbb{R}, \tag{4.7}$$

$$\lim \sqrt{r}\,(\partial_r - ik)\,u = 0 \qquad \text{as } r = |\mathbf{x}| \to \infty. \tag{4.8}$$

[1]Bounded in $L^2(\mathbb{R})$ for any fixed y, or between any pairs of Sobolev spaces $H^{s_1}(\mathbb{R})$ and $H^{s_2}(\mathbb{R})$ as Fourier analysis would show.

for $\kappa > 0$. The solution of this problem is expressed via Fourier analysis as

$$u(x, y) = \frac{1}{(2\pi)^2} \int_{\mathbb{R}} e^{ix\xi} e^{iy\sqrt{\kappa^2 - \xi^2}}\, \widehat{u}_0(\xi)\, d\xi, \tag{4.9}$$

where the branch cuts of the square root respectively extend to $\pm i\infty$ at $\xi = \pm\kappa$, so as to respect the outgoing boundary condition [?]. The expression of the resulting distributional Poisson kernel is of no concern, but let us continue to call it $P_y(x)$.

The oscillatory content of u_0 determines the type of waves that result from solving the Helmholtz equation, namely:

- when $|\xi| < \kappa$, the waves are propagating or traveling, with horizontal wave vector $\xi_x = \xi$ and vertical wave vector $\xi_y = \sqrt{\kappa^2 - \xi^2}$;
- when $|\xi| > \kappa$, the waves are evanescent, and decay exponentially in $y > 0$. The limiting case $|\xi| = \kappa$ corresponds to grazing waves.

The interesting situation for stable extension from data in an interval is when u_0 is a priori known to only generate propagating waves, i.e., when the integral in (4.9) is restricted to $[-\kappa, \kappa]$. Additionally, the waves of interest will be further restricted to propagate in directions well away from those of the grazing waves ($\xi = \pm\kappa$), by means of an extra number c that indicates the bandlimit of the boundary data u_0.

Definition 12. *We call cone-directed wave with aperture c/κ any solution of the Helmholtz equation in $\mathbb{R} \times \mathbb{R}_0^+$ of the form (4.9), with the restriction that*

$$\mathit{supp}\, \widehat{u_0} \subset [-c, c]$$

for some $c \leq \kappa$.

For notational convenience, it is advantageous to let $\xi = ct$, with $-1 \leq t \leq 1$, and write (4.9) as a so-called Herlgotz wave

$$u(x, y) = H[h](x, y) = \int_{-1}^{1} e^{icxt} e^{iy\sqrt{\kappa^2 - c^2t^2}}\, h(t)\, dt, \tag{4.10}$$

with $h(t)$ the Herglotz density. Note that for $c = \kappa$, Herglotz waves are dense in the space of solutions of the Helmholtz equation in any finite set [42]. In the half-space, equation (4.10) defines waves that satisfy the Sommerfeld radiation condition when $h(t)$ has a bounded L^2 norm, or may not when h is distributional (e.g. plane waves). In the sequel, we assume $c < \kappa$.

The bicharacteristics (rays) emanating from points on the segment γ, and oriented with wave vectors admissible in the sense of the definition above, form a *cone of influence* opening up, with angle $2\arcsin(c/\kappa)$. See figure 4-1. If u_0 were essentially compactly supported inside γ, then the solution of the Helmholtz equation would be essentially compactly supported inside the cone of influence — an assumption that we do not make in this paper. More important for us is the *cone of dependence*, defined as the complement of the cone of influence for the complement of γ.

Definition 13. *Let $\gamma = [-x^*, x^*]$. We call cone of dependence with aperture c/κ, and denote by $\Gamma(c/\kappa)$, or simply Γ, the set*

$$\left\{ (x,y) \in \mathbb{R} \times \mathbb{R}_0^+ : |x| \leq x^* - \frac{y}{\sqrt{(\kappa/c)^2 - 1}} \right\}.$$

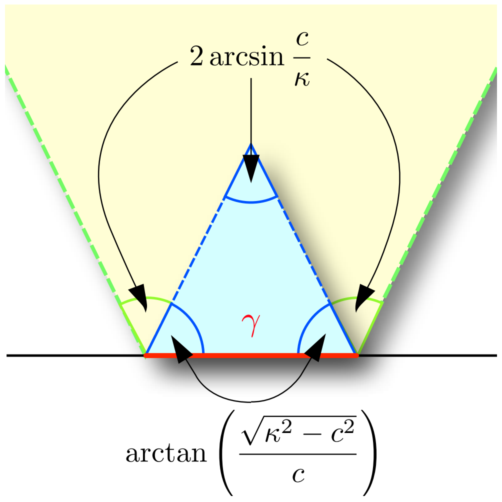

$2 \arcsin \frac{c}{\kappa}$
γ
$\arctan\left(\frac{\sqrt{\kappa^2 - c^2}}{c}\right)$

Figure 4-1: Dependence cone (light blue) and influence cone (yellow) of γ (red), for a given aperture c/κ.

In this Chapter we plan to argue that the extrapolation is stable in the cone of influence. Theorem 3 is part of the answer in which we can observe the importance of the cone of influence.

4.3 Stability with respect to the Herglotz density

The stability theory is very favorable if $u(x, y)$ is seen as determined by the Herglotz density h, even when h is a distribution in a negative Sobolev space. Since h is a rescaled version of the Fourier transform $\widehat{u_0}(\xi)$, its knowledge is equivalent to the whole of $u_0(x)$ without restriction on $x \in \mathbb{R}$.

Theorem 3. *Let u be given by (4.10), interpreted as a duality pairing of the exponential kernel with $h \in W^{-s,p}(-1,1)$ for some $s \geq 0$ and some $1 < p \leq \infty$. Then*

$$|u(x,y)| \leq C_{p,s} \left(1 + |x| + \frac{y}{\sqrt{\kappa^2/c^2 - 1}} \right)^s \|h\|_{W^{-s,p}}. \tag{4.11}$$

Theorem 3 is proved in Appendix 4.A. We would have wished to have a proof that involves a weighted Sobolev norm of $u_0(x)$ in the right-hand side of Eq. 4.11, but we were unable to obtain such result.

The geometry of the cone of dependence Γ supported by $\gamma = [-x^*, x^*]$ is apparent in Thm. 3. In the region of the half-plane where $y \leq x^* \sqrt{\kappa^2/c^2 - 1}$ (i.e., when y is smaller than the height of the cone Γ), and when (x, y) is outside Γ, then

$$|x| + \frac{y}{\sqrt{\kappa^2/c^2 - 1}} = x^* + \operatorname{dist}\left((x, y), \Gamma\right),$$

where dist denotes the horizontal distance of a point to a set.

The result indicates that extension is not only possible, but completely stable, *as a function of h*. The region of stability is either the whole half-plane when h is an L^p function, or is reduced to a (soft) neighborhood of the dependence cone in the case when h is a distribution. Finally, we point out that Thm. 3 is easy to extend to $s < 0$.

4.3.1 Truncation in the PSWF domain

In practice, extrapolation needs to be robust in the presence of truncation to a finite-size problem. This level of robustness in not directly accessible from Thm. 3, since a truncation error on $u|_\gamma$ can not easily be modeled through the density h. In the next 3 Sections, we aim to provide quantitative arguments that support the robustness under truncation in a specific system, the prolate spheroidal wave functions (PSWF), which are the eigenfunction of the operator linking the Herglotz density h to the partial data $u|_\gamma$.

In this section, we first establish convergence of the truncation for finite N of the boundary data $u|_\gamma$ using the PSWF (Eq. 4.39),then we show the convergence of the truncated Herglotz density, when the later possesses some degree of smoothness (Lemma 9), and, finally, we provide a formula to extrapolate the truncated boundary data (Eq. 4.21). The details can be found in the proof of Lemma 9 in the Appendix 4.A. The proof highlights the physics of wave extrapolation by transforming the extrapolation problem using a projection onto a truncated PSWF expansion.

It is possible to build a tight truncated inverse using the PSWF, whose theory was developed in the 1970's when Slepian, Pollack and Landau were investigating the concentration problem of band-limited functions. They studied the operator $F_c^* F_c$, where F_c is a bandlimited Fourier transform. Their most striking observation was that the Laplacian written in prolate spheroidal coordinates commutes with this operator. This observation led to the series of seminal papers [123, 89, 90, 121, 122].

The prolate spheroidal wave functions (PSWF) are defined as the eigenvalues of a band-limited Fourier transform, or alternatively as the eigenvalues of a Sturm-Liouville operator; furthermore, they solve an optimization problem.

Definition 14. *Let $c > 0$. Define the operator $F_c : L^2([-1, 1]) \to L^2([-1, 1])$ by*

$$F_c[\varphi](x) = \int_{-1}^{1} e^{icxt} \varphi(t) dt.$$

The operator F_c is compact and its eigenfunctions form an orthonormal basis

$\{\psi_n^c\}_{n=1}^\infty$ of $L^2([-1,1])$ with associated eigenvalues $\{\lambda_n^c\}_{n=0}^\infty$ decaying to zero and $\{\lambda_k^c \psi_n^c\}_{n=1}^\infty$ form an orthonormal basis of $L^2(\mathbb{R})$ (see Theorem 2.4 in [146]).

By definition, $H[h](x,0) = F_c[h](x)$, which allows us to build a truncated inverse by

$$H_N^{-1}u(t) = \sum_{k=0}^{N} \frac{\alpha_k}{\lambda_k} \psi_k^c(t), \qquad \alpha_k = \int_{-1}^{1} u(x)\psi_k^c(x)dx. \tag{4.12}$$

Using the approximated inverse and the smoothness of the Herglotz density we can recover the density, and extrapolate the wavefield to the whole half-plane.

Lemma 9. *Let $\kappa > c$, $s > 3/2$, $\gamma = [-1,1] \times \{0\}$, and let $u = H[h]$ with $h \in H^s([-1,1])$, then it is possible to recover h from Dirichlet data on γ, i.e.,*

$$\left\| h - H_{0,N}^{-1} u|_\gamma \right\|_{L^2([-1,1])} \leq \frac{1}{2} \sum_{k=N+1}^{\infty} C \left(k^{-\frac{2}{3}s} \|h\|_{H^s[-1,1]} + q_k^{\delta k} \|h\|_{L^2[-1,1]} \right), \tag{4.13}$$

where $C > 0$ and $\delta > 0$ are constants independent of k, and $q_k = \sqrt{\frac{c^2}{\lambda_k^2}} < 1$ for k large.

As a consequence, if we combine Lemma 9 with Thm. 3, then we have that

$$\|H[h] - H[H_{0,N}^{-1} u|_\gamma]\|_{L^\infty(\mathbb{R}_+^2)} < \epsilon \|h\|^{H^s[-1,1]}, \tag{4.14}$$

provided that $s > 3/2$.

Algorithmically, using the proof of Lemma 9 and the properties of the PSWF, we can write the extrapolation in the truncated setting as

$$\begin{aligned}
H[H_{0,N}^{-1} u|_\gamma](x,y) &= \int_{-1}^{1} \sum_{k=0}^{N} \frac{\alpha_k}{\lambda_k^c} e^{icxt} e^{iy\sqrt{\kappa^2 - c^2 t^2}} \psi_k^c(t) dt, && (4.15)\\
&= \sum_{k=0}^{N} \frac{\alpha_k}{\lambda_k^c} \int_{-1}^{1} e^{icxt} e^{iy\sqrt{\kappa^2 - c^2 t^2}} \psi_k^c(t) dt, && (4.16)\\
&= \sum_{k=0}^{N} \frac{\alpha_k}{\lambda_k^c} \int_{-1}^{1} e^{icxt} e^{iy\sqrt{\kappa^2 - c^2 t^2}} \int_{\mathbb{R}} e^{-ictz} \lambda_k^c \psi_k^c(z) dz dt, && (4.17)
\end{aligned}$$

using the definition of the PSWF and its extension to the real axis via the Fourier transform (see [123]). In addition, formally we have that

$$H[H_{0,N}^{-1}u|_\gamma](x,y) = \sum_{k=0}^{N} \frac{\alpha_k}{\lambda_k^c} \int_{\mathbb{R}} \int_{-1}^{1} e^{ic(x-z)t} e^{iy\sqrt{\kappa^2 - c^2t^2}} \lambda_k^c \psi_k^c(z) dz dt, \tag{4.18}$$

$$= \sum_{k=0}^{N} \frac{\alpha_k}{\lambda_k^c} \int_{\mathbb{R}} \left(\int_{-1}^{1} e^{ic(x-z)t} e^{iy\sqrt{\kappa^2 - c^2t^2}} dt \right) \lambda_k^c \psi_k^c(z) dz, \tag{4.19}$$

$$= \sum_{k=0}^{N} \frac{\alpha_k}{\lambda_k^c} \int_{\mathbb{R}} K_y^c(x-z) \lambda_k^c \psi_k^c(z) dz, \tag{4.20}$$

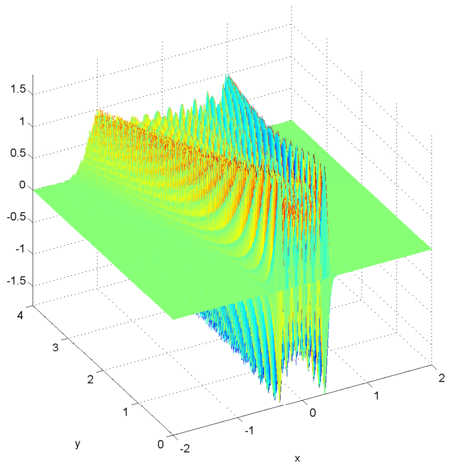
$$= \sum_{k=0}^{N} \frac{\alpha_k}{\lambda_k^c} K_y^c * \lambda_k^c \psi_k^c(x). \tag{4.21}$$

This expression allows to decouple the extrapolation kernel K_y^c from the projection of u_0 onto the PSWF.

4.3.2 Size properties of the PSWF and the extrapolation kernel

Although we would have liked to have a stability result with respect to the partial data, as would be implied by a weighted Sobolev norm of u_0 in the right-hand side of Eq 4.11, we were unable to obtain such a result. However, we have quantitative (non-rigorous) arguments and extensive numerical experiments that seem to confirm that the extrapolation can be performed in a stable manner with respect to u_0, even for h rougher than $W^{-s,p}$. We explain this behavior by the decoupling between the regularity of h, which is characterized by the decay rate of $\alpha_k = \langle u|_\gamma, \psi_k^c \rangle$, and the extrapolation of the PSWF, which is given by the convolution with the extrapolation kernel $K_y^c(x)$ (see Eq. 4.21) defined in Lemma 12.

Lemma 12 shows that this convolution kernel is concentrated at the origin in a length scale that depends linearly on y; this observation leads to the dependency cone and the influence cone as presented in Fig. 4-1. The influence cone is the region of the space in which most of the energy present on γ is radiated to, whereas the dependence (or extrapolation) cone is the region in which most of the energy present is radiated only from γ.



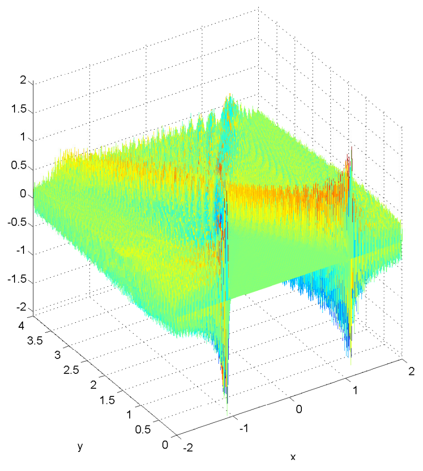


Figure 4-2: Extrapolation of two different PSWF, for $\kappa = 60\pi$, $\kappa/c = 3/2$, left: ψ_{10}^c, right: ψ_{100}^c.

From an analytical point of view, the influence cone arises from the fact that the bandwidth of the PSWF is included in $[-c, c]$ and $c < \kappa$, which removes the singularity in the phase of $e^{ict}e^{\sqrt{(\kappa/c)^2-t^2}}$, converting it in a smooth kernel (via a Fourier transform) with fast decay (up to a smooth cut-off). The decay of the convolution kernel is characterized by a length-scale that is linear in y, for y larger than the characteristic wave-length of the data. The extrapolation cone is due primarily to the combination of the growth properties of the PSWF and the concentration of the convolution kernel at the origin.

The convergence of the truncated sum in Eq. 4.21 depends on the growth properties of α_k. It can be shown that if the associated Herglotz density is a distribution, then the ratio $\frac{\alpha_k}{\lambda_k^c}$ may not decay. However, numerically, we observe that $(K_y^c * \lambda_k^c \psi_k^c)(x)$ decays fast when (x, y) lies in the extrapolation cone, then providing local convergence inside the cone.

Intuitively, for small n, the PSWF are concentrated inside $[-1, 1]$, hence when extrapolated (i.e. convolved with the extrapolation kernel for different y) they will radiate from there as in Fig 4-2 (left). However, as n increases the PSWF will have less energy in this segment; henceforth, their extrapolation will radiate from outside this interval as depicted in Fig. 4-2 (right). It is possible to prove that for large n, $\lambda_n^c \psi_n^c$ are bounded in $L^\infty(\mathbb{R})$ and are factorially small in $[-1, 1]$ (see Theorem 33 in [105]), then it experiments an exponential grow, and finally, it decays as $1/|x|$ as $|x|$ tends to infinity (see Lemma 10).

Lemma 10. *It exists $n^* > 2c/\pi$, such that $\forall n > n^*$ the following properties hold,*

1. $|\psi_n^c(x)| \leq 2\sqrt{n}$ *for* $|x| < 1$,

2. $\psi_n^c(x) \lesssim \frac{1}{\lambda_n^c c x}$,

3. $\|\lambda_n^c \psi_n^c\|_{L^\infty(\mathbb{R})} \lesssim c$, *independently of* n.

Numerically, we have that PSWF exhibits two “humps”, one in the $\mathbb{R}^+$ and another in $\mathbb{R}^-$, which move away from zero as n increases. As stated before, the extrapolation kernel has most of its mass concentrated at zero. For n large the fast decaying tail of the kernel hits the hump that is far away from the origin; on the other hand, the hump of the kernel hits a factorially small PSWF close to the origin. This implies that the extrapolated PSWF becomes small inside the extrapolation cone as n increases.

Lemma 11. *Let $\kappa/2c > \alpha > 1$, and let $\eta_\alpha \in \mathcal{C}_c^\infty(\mathbb{R})$ such that*

$$\eta_\alpha(t) = \begin{cases} 1 & \text{if } |t| < 1 \\ 0 & \text{if } |t| > \alpha \end{cases}. \tag{4.22}$$

Then

$$\left|\partial_t^n e^{icy\sqrt{(\kappa/c)^2-t^2}}\eta_\alpha(t)\right| \leq C_{\alpha,n} c_y^n, \qquad \text{with } c_y = \max\left(\frac{c^2 y}{\sqrt{\kappa^2-c^2}}, 1\right). \tag{4.23}$$

Lemma 12. *Let $\kappa/c > 2$, $\kappa/2c > \alpha > 1$, $\eta_\alpha(t)$ be the smooth-cut off in Eq. 4.22, and let*

$$K_y(x) = \int_{-\alpha}^{\alpha} e^{ixt} e^{iy\sqrt{\kappa^2-c^2t^2}}\eta_\alpha(t)dt, \tag{4.24}$$

be the extrapolation convolution kernel. Then,

$$|K_y^c(x)| = |K_y(cx)| \leq C_n \left(1 + \frac{c|x|}{c_y}\right)^{-n} \qquad \forall n \in \mathbb{N}, \qquad \text{with } c_y = \max\left(\frac{c^2 y}{\sqrt{\kappa^2-c^2}}, 1\right). \tag{4.25}$$

4.3.3 Extrapolation Error

In this section we use the observations and results from earlier Sections in this Chapter to provide a formal analysis of the extrapolation error in an idealized situation.

Fix an error ϵ, a band-limit c, and a frequency κ, such that $\kappa > c$. Let $u|_\gamma$ be the restriction to γ of $u_0(x)$, where $u_0(x) = u(x,0) = H[h](x,0)$, with aperture c.

Suppose that we are only able to measure the noisy data

$$v|_\gamma = u|_\gamma + \delta u|_\gamma, \tag{4.26}$$

in which we suppose that

$$\delta\alpha_k = \langle \delta u|_\gamma, \psi_k^c \rangle = \mathcal{O}(\epsilon^2) \tag{4.27}$$

Let $N > 2c/\pi + 10$ be large enough, such that $\lambda_{N+1}^c \sim \epsilon$. In general, we have that $N = \mathcal{O}(\log(1/\epsilon))$ (Thm. 2.5 in [146]).

Using Eq. 4.21 we have that

$$v^{\text{extrp}}(x,y) = H[H_N^{-1}v|_\gamma](x,y) = \sum_{k=0}^{N} \frac{\alpha_k + \delta\alpha_k}{\lambda_k^c} K_y^c * \lambda_k^c \psi_k^c(x) \tag{4.28}$$

Then the extrapolation error is given by

$$(v^{\text{extrp}} - u)(x,y) = \sum_{k=0}^{N} \frac{\delta\alpha_k}{\lambda_k^c} K_y^c * \lambda_k^c \psi_k^c(x) - \sum_{k=N+1}^{\infty} \frac{\alpha_k}{\lambda_k^c} K_y^c * \lambda_k^c \psi_k^c(x). \tag{4.29}$$

Now, using $\|\lambda_k^c \psi_k^c\| = \mathcal{O}(c)$ and $|K_y^c(x)| = \mathcal{O}(1)$, we can bound

$$\left| \sum_{k=0}^{N} \frac{\delta\alpha_k}{\lambda_k^c} K_y^c * \lambda_k^c \psi_k^c(x) \right| < CN\epsilon = \mathcal{O}(\epsilon \log(1/\epsilon)) \tag{4.30}$$

For the remaining term, we have that $\frac{\alpha_k}{\lambda_k^c} = \mathcal{O}(1)$ if h is rough (for example, a Dirac delta).

Using the arguments explained in the Section 4.3.2, in which we argued that the for large n we have that $K_y^c * \lambda_k^c \psi_k^c(x) = \mathcal{O}(c\lambda_k^c)$ within the cone of influence. This is because the absolute value of extrapolated PSWF within the cone of influence only depends on the energy contained in γ, which in this case is given by the norm of $\lambda_k^c \psi_k^c(x)$ when $x \in [-1,1]$ that is $\mathcal{O}(c\lambda_k^c)$ (from Lemma 10).

Moreover, given that we supposed that N is already in the factorially decaying regime (Thm. 2.5 in [146]) we have that

$$\left| \sum_{k=N+1}^{\infty} \frac{\alpha_k}{\lambda_k^c} K_y^c * \lambda_k^c \psi_k^c(x) \right| < C \left| K_y^c * \lambda_{N+1}^c \psi_{N+1}^c(x) \right| < C\epsilon \tag{4.31}$$

for x and y within the cone of influence.

Finally, we have that

$$\left| (v^{\text{extrp}} - u)(x,y) \right| = \mathcal{O}(\epsilon \log(1/\epsilon)) \tag{4.32}$$

provided that (x,y) lies inside the influence cone.

4.4 Numerical Examples

In this section, we illustrate our observations with numerical examples. We start by showing the extrapolated PSWF in the cases when $n < 2c/\pi$ and $n > 2c/\pi$. Next, we depict the behavior of the extrapolation of a Herglotz wave with a smooth density. We point out that it is not compulsory to use the PSWF to ensure an accurate extrapolation; other cheaper and faster procedures are available to perform a, seemingly, stable extrapolation. Finally, we show examples of some simple but high frequency solutions for the Helmholtz equation obtained using this stable extrapolation technique.

4.4.1 Extrapolation

To illustrate the theoretical results, we present some numerical experiments, in which we discretized the operators H and H_0 using an ad-hoc quadrature. The quadrature used is adapted to the PSWF and it was developed by Boyd [22]. Let $\{w_\ell\}_{\ell=0}^{N_s}$ be the Gauss-Lobatto quadrature weights and $\{t_\ell\}_{\ell=0}^{N_s}$ the quadrature points. Thus

$$\langle \boldsymbol{\psi}_n^c, \boldsymbol{\psi}_m^c \rangle_w := \sum_{\ell=0}^{N_s} \psi_n^c(x_\ell) w_\ell \psi_m^c(x_\ell) = \langle \psi_n^c, \psi_m^c \rangle_{[-1,1]}, \tag{4.33}$$

up to machine precision for n and $m < N_s$, where the number of samples N_s is greater than the number of samples given by the Shannon-Nyquist sampling rate, but of the same order of magnitude.

The extrapolation operator, H is factorized as $H_0 \circ m_y$, where H_0 is a band-limited Fourier Transform and m_y is just a function multiplication. H is defined on $L^2[-1,1] \to \mathcal{C}(\mathbb{R}) \subset L^2(\mathbb{R})$. To discretize it, we define the set of evaluation points that are given by $\{x_j\}_{j=0}^{N_e}$. The other operators are defined using their discrete counterparts given by the quadrature in Eq. 4.33,

$$\boldsymbol{u}_\ell = u(t_\ell), \qquad (\boldsymbol{H_0})_{k,\ell} = w_\ell e^{icx_k t_\ell}, \qquad (\boldsymbol{m_y})_{k,\ell} = \delta_{\ell,k} e^{icy\sqrt{(\kappa/c)^2 - c^2 t_\ell^2}}. \tag{4.34}$$

To compute $\boldsymbol{H}_{0,N}^{-1}$, we use the SVD regularized inverse of $\boldsymbol{H_0}$, tuned for a precision of 12 digits, i.e.,

$$\mathbf{H_{0,N}^{-1}} = \operatorname{diag}\left(\left\{ \frac{1}{\lambda_k} \right\}_{k=0}^{N} \right) \cdot \left[\boldsymbol{\psi}_0^c, \boldsymbol{\psi}_1^c, \dots, \boldsymbol{\psi}_{N-1}^c, \boldsymbol{\psi}_N^c \right] \cdot \operatorname{diag}(\{w_\ell\}_{\ell=0}^{N_s}). \tag{4.35}$$

Therefore, given the trace $u|_\gamma = v$, the extrapolation at the points $\{x_j\}_{j=0}^{N} \times \{y\}$ is given by

$$\{u(x_j, y)\} = \mathbf{u}_y = \mathbf{H_0} \cdot \mathbf{m_y} \cdot \mathbf{H_{0,N}^{-1}} \cdot \mathbf{v}. \tag{4.36}$$

The waves generated by extrapolating the PSWF are illustrated in Fig. 4-2. The extrapolation of a PSWF with $n < 2c/\pi$ is depicted in Fig. 4-2 *(left)*; in which the wave radiates from the interior of the segment $[-1,1]$. Fig. 4-2 (*right*) shows a typical example of an extrapolated wave when $n > 2c/\pi$, in which the energy is being radiated from outside the segment $[-1,1]$, resulting in a cone in which the absolute value of the extrapolated wave is small. Fig. 4-3 depicts the cone in which the solution is small for different ratios κ/c. In addition, Fig. 4-3 shows the theoretical scaling for the cone and its slope which is given by $\frac{c}{\sqrt{\kappa^2 - c^2}}$. The theoretical slope given by Thm. 3 coincides with the slope provided by the numerical experiments.

Fig. 4-4 presents the extrapolation of a Gaussian beam, which is smooth and essentially band-limited. The figure shows that the extrapolation is valid in more than 2000 wavelengths from the origin.

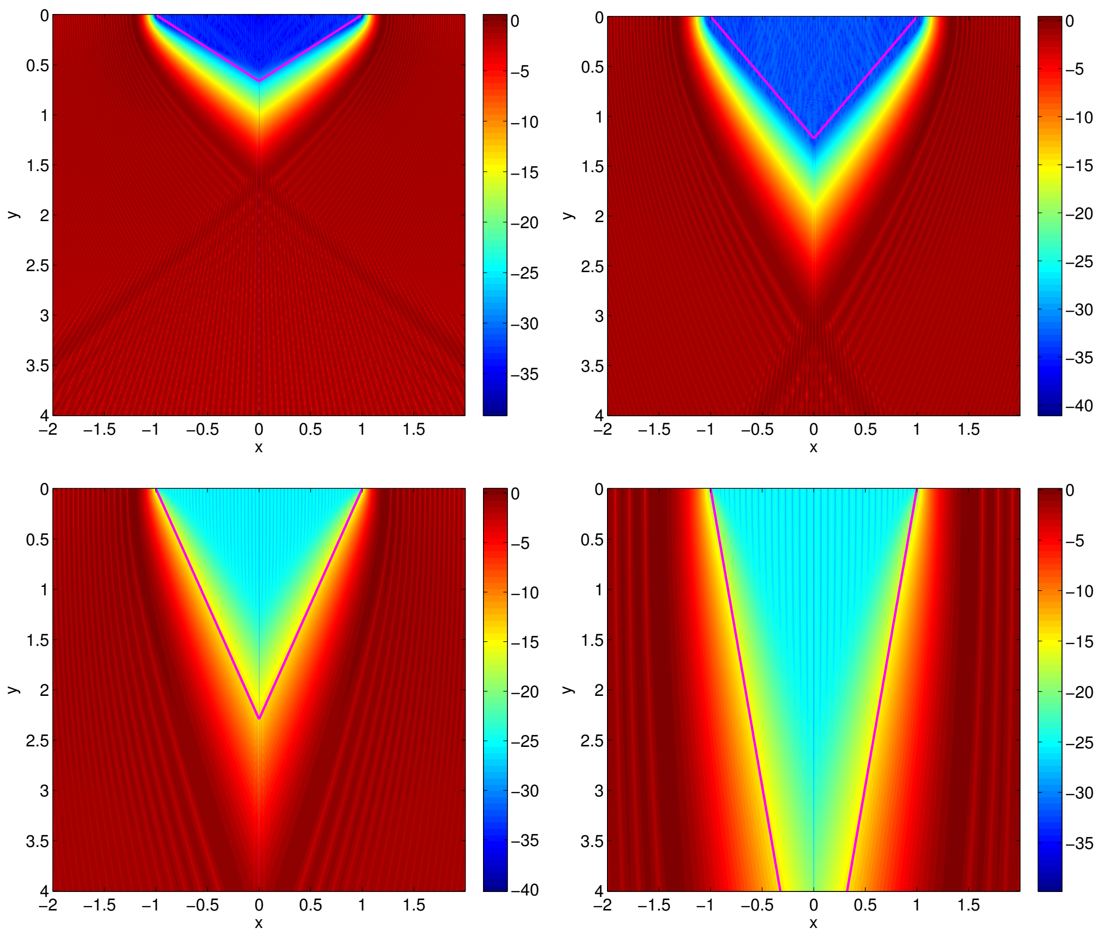


Figure 4-3: Absolute value of the extrapolation of PSWF for different ratios κ/c, $\kappa = 60\pi$. *Top left*: $\kappa/c = 6/5$; *top right*: $\kappa/c = 5/3$; *bottom left*: $\kappa/c = 5/2$; *bottom right*: $\kappa/c = 6$. The boundaries of the extrapolation cone from Thm. 3 are drawn in each picture (in purple).

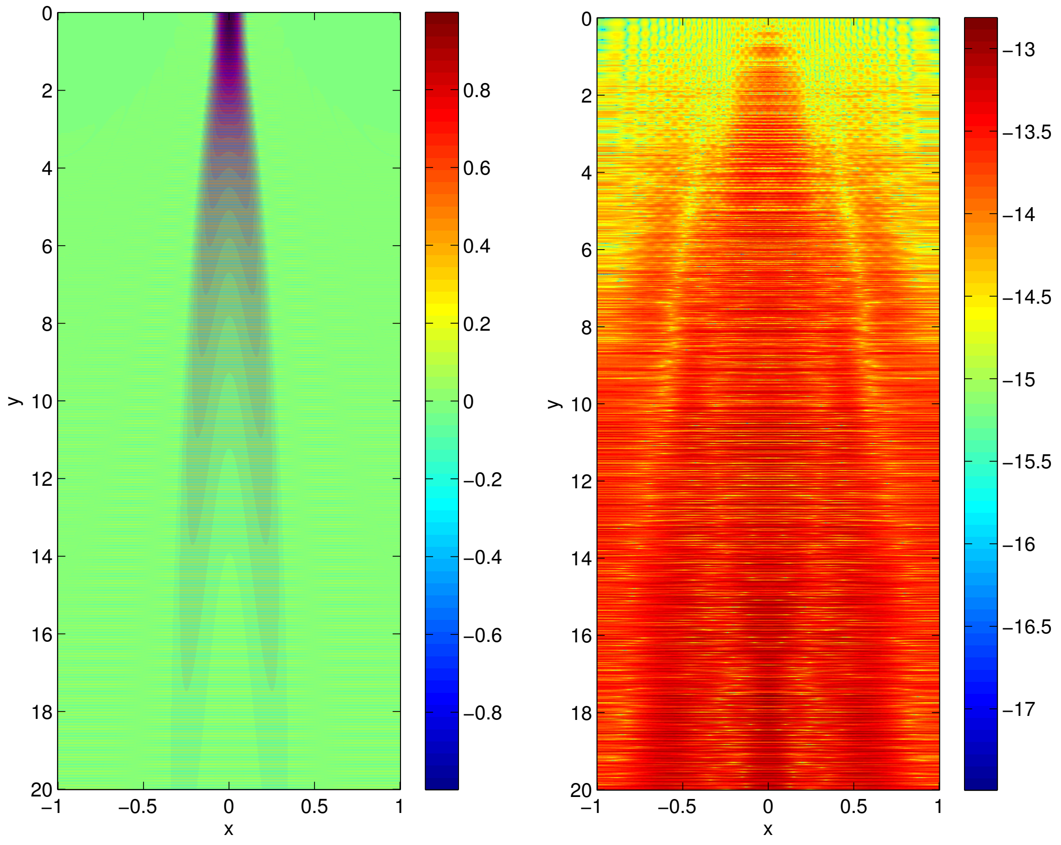


Figure 4-4: Extrapolation of a Gaussian beam. *Left*: analytical solution; *right*: error in log scale (base 10) of the error of the extrapolation $\kappa = 50\pi$.

Fig. 4-5 shows the same procedure with a Herglotz density that is a sum of random plane waves. The error is shown in Fig. 4-5 *(left)* that depicts the cone of extrapolation given by the theory, which is supported on $[-1, 1]$.

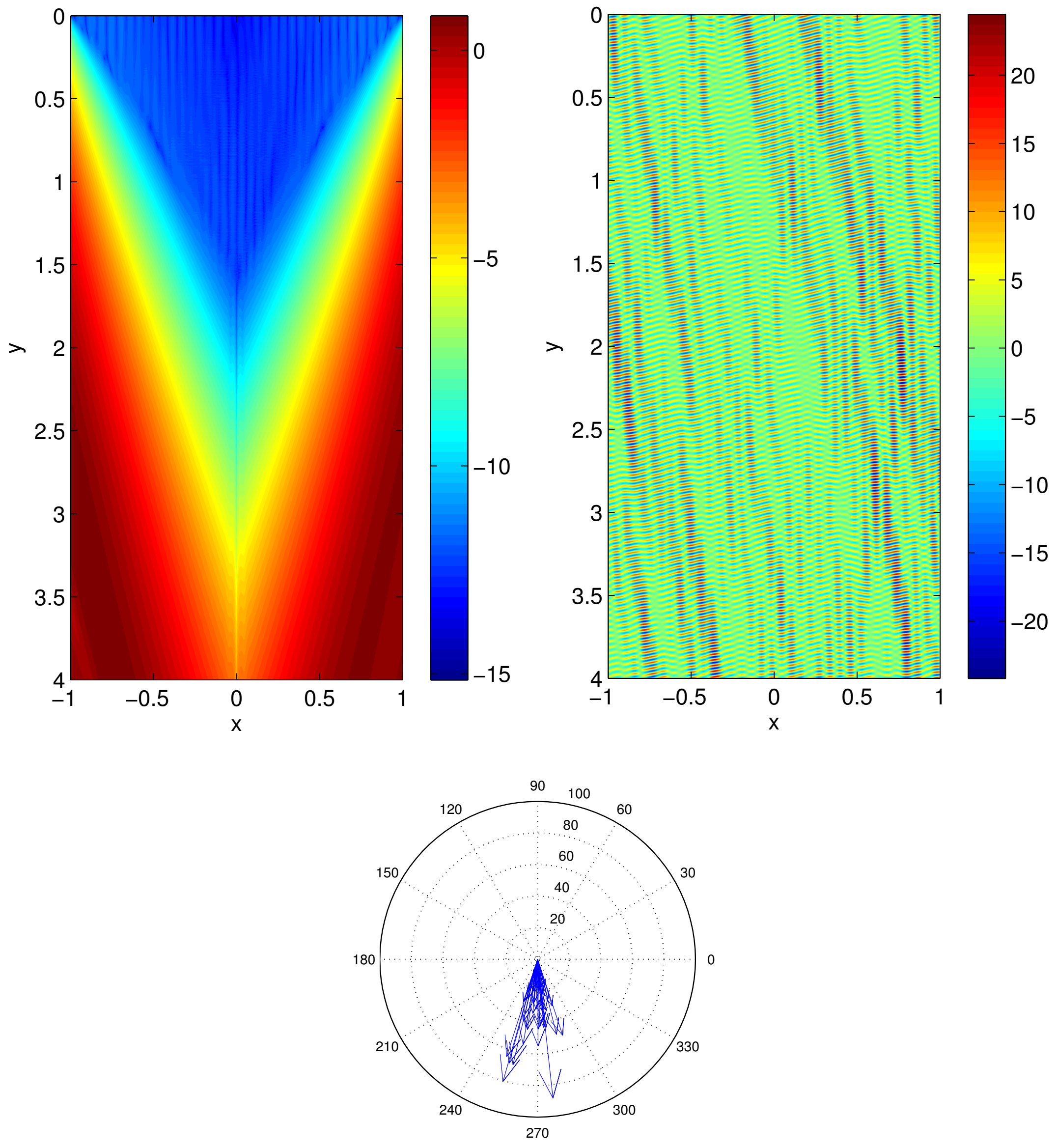


Figure 4-5: Extrapolation of 70 random complex exponentials for $\kappa = 60\pi$ and $\kappa/c = 3$. *Top left*: analytical solution; *top right*: error in log scale (base 10) for the error of the extrapolation; *bottom* the direction and magnitude of the 70 different plane waves.

4.4.2 Pivoted QR and broken lines

The numerical experiments presented in the previous subsection provide strong evidence that the extrapolation technique using PWSF is stable and accurate; however, building the quadrature points and weights is computationally expensive. It uses a recurrence relation via Legendre polynomial to obtain the PSWF and then an optimization routine based on a Newton iteration to get a very precise quadrature rule. We found that using a regular discretization on a segment plus a naive discretization of the integral leads to the same qualitative results, which is, in addition, trivially extended to broken lines.

We discretize the Herglotz operator using the simplest quadrature rule possible. Let γ be a straight line and $\{(x_i, y_i)\}_{j=1}^{N_s} \subset \gamma$ be a set of equidistant points contained in γ. In addition, let $\{\theta_j\}_{j=1}^{N_\theta} \subset [-\arccos(\kappa/c), \arccos(\kappa/c)]$ be a regular angular discretization for θ. We discretize the Herglotz operator by

$$(\mathbf{H}_{\mathbf{0},\gamma})_{i,j} = \frac{1}{\Delta\theta} e^{ik(x_i \sin\theta_j + y_i \cos\theta_j)}. \tag{4.37}$$

Even though the error obtained using the naive quadrature is slightly larger than the error given by constructing the PSWF, the overall computational cost is greatly reduced. Moreover, using this simple quadrature, this extrapolation method can be easily generalized to work with broken lines. Given different connected segments $\{\gamma_\ell\}_{\ell=1}^{L}$, we define the operators $H_{0,\gamma_\ell} : g(\mathbb{S}^1) \to L^2(\gamma_\ell)$. For given Dirichlet data on several segments, it is possible to recast the problem of estimating the Herglotz density as a linear algebra problem:

$$[u|_{\gamma_1}, u|_{\gamma_2}, \dots, u|_{\gamma_{L-1}}, u|_{\gamma_L}]^t = [H_{0,\gamma_1}, H_{0,\gamma_2}, \dots, H_{0,\gamma_{L-1}}, H_{0,\gamma_L}]^t g. \tag{4.38}$$

Algorithm 16. *Broken line*

```
1: procedure DENSITY ESTIMATION($\{\gamma_\ell\}_{\ell=1}^{N_\ell}, \{\boldsymbol{u}|_{\gamma_\ell}\}_{\ell=1}^{N_\ell}$)
2:     for $j < N_\ell$ do                                          ▷ Build the operator for each $\gamma_\ell$
3:         $(\mathbf{H}_{\mathbf{0},\gamma_\ell})_{i,j} = \frac{1}{\Delta\theta} e^{ik(x_i \sin\theta_j + y_i \cos\theta_j)}$          ▷ $(x_j, y_j) \in \gamma_\ell$
4:     end for
5:     $\mathbf{H} = [\mathbf{H}_{\mathbf{0},\gamma_1}; \mathbf{H}_{\mathbf{0},\gamma_2}; \dots; \mathbf{H}_{\mathbf{0},\gamma_{N_\ell}}]$                        ▷ Concatenate the operators
6:     $\mathbf{u} = [\boldsymbol{u}|_{\gamma_1}^t, \boldsymbol{u}|_{\gamma_2}^t, \dots, \boldsymbol{u}|_{\gamma_{N_\ell}}]^t$                           ▷ Concatenate the data
7:     $\mathbf{g} = \mathbf{H} \backslash \mathbf{u}$                                          ▷ Use MATLAB's backslash
8:     return g
9: end procedure
```

In which Eq. 4.38 is discretized using Eq. 4.37. To solve this exponentially-ill conditioned discretized system, we found that MATLAB's backslash operator, which relies on a pivoted QR factorization (see Lecture 4 in [137]), was the fastest and most accurate option to solve the linear system given by Eq. 4.38. In normal circumstances we would need to compute the pseudo-inverse; however, the pseudo-inverse is limited by the conditioning number of the matrix that is very large because the continuum operator is compact. In fact, as the solution is not unique (numerically), we have the freedom of choosing the most convenient one. In this specific case, the solution given by the pivoted QR factorization in MATLAB's backlash operator had the smallest residual. Alg. 16 summarizes the extrapolation procedure. Moreover, Fig. 4-6 presents some examples in which the solution is accurate inside a cone supported on the broken line.

Remark 9. *We point out that using a naive quadrature has the same qualitative* *behavior for the extrapolation of singular Herglotz densities. The extrapolation of a Herglotz wave with a smooth density using the naive quadrature will no longer result in a global extrapolation. In fact, the extrapolation will be accurate only in a cone with the same slope as that in the singular case.*

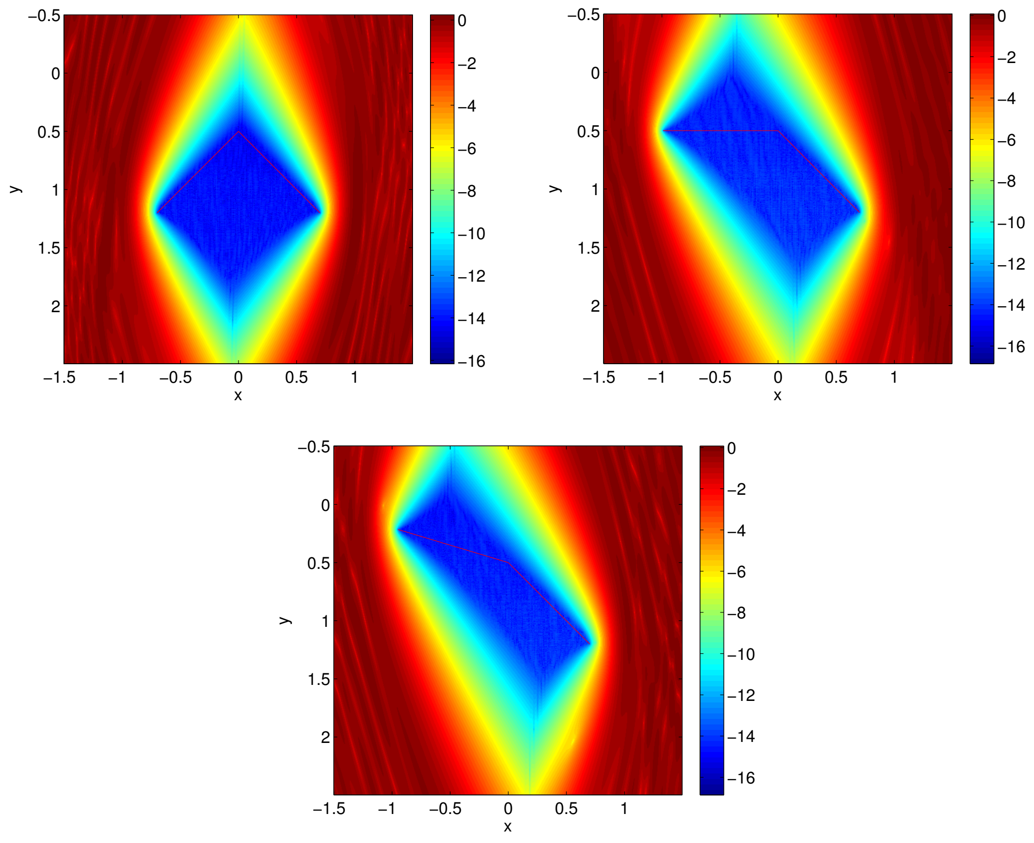


Figure 4-6: Error of the extrapolation of 70 random complex exponentials from a broken line, for $\kappa = 60\pi$ and $\kappa/c = 2$. We show different configurations of γ_1, γ_2 (in purple.)

4.4.3 Towards an efficient Helmholtz solver

Based on the results of the previous sections, we developed a simple yet accurate sweeping-like solver for the Helmholtz equation in a homogeneous medium. Using the extrapolation for broken lines, it is possible to sweep the domain, with some geometrical constraints, building the solution from the boundary towards the interior of the domain. It is trivial to adapt the method from broken lines to the edges of a triangular mesh. The method is analogous to solving a discontinuous Galerkin or Trefftz formulation, using plane waves as basis functions; however, instead of inverting a large ill-conditioned linear system, we solve the unknowns starting from the the Dirichlet boundary towards the interior of the domain in an ordered manner. This

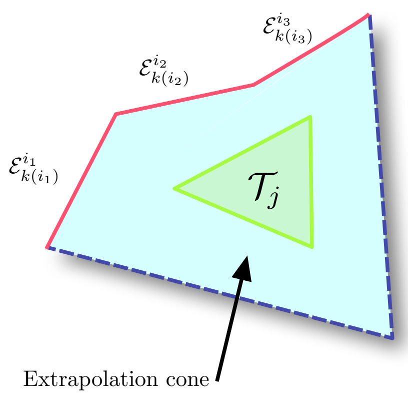

$\mathcal{E}^{i_3}_{k(i_3)}$
$\mathcal{E}^{i_2}_{k(i_2)}$
$\mathcal{E}^{i_1}_{k(i_1)}$
$\mathcal{T}_j$
Extrapolation cone

Figure 4-7: Broken line configuration to ensure accuracy for the extrapolation.

process can be seen as the analytical counterpart of the sweeping methods (see [59]), in which the elimination of unknowns is replaced by the extrapolation procedure. This extrapolation procedure uses only the degrees of freedom on the edges of the triangles.

Let $\{\mathcal{T}_j\}_{j=1}^{N_T}$ be a set of triangles of a triangular mesh of a bounded domain Ω. We define an approximate Herglotz density, g_j, in each triangle. For each triangle $\mathcal{T}_j$, we note its edges as $\{\mathcal{E}_j^i\}_{i=1}^3$, which are discretized with equispaced points.

The sweeping method consists of several passes; for each one of those, we need to choose a sweeping direction and an aperture. Each sweep consists in updating the Herglotz density in a particular triangle using upwind information, i.e., information from the already updated triangles. The new Herglotz density is then used to update the Herglotz densities of the triangles not yet updated.

To update a particular triangle, we need a set of edges that forms a broken line, such that its dependence, or extrapolation, cone (defined in Def. 13) contains the triangle to be updated (see Fig. 4-7.) The set of edges must be contained in the triangles already updated. We sample the local information from the triangles to their edges using Alg. 17. Then we use the information on the edges to estimate the Herglotz density and update it. This procedure is written in pseudo code in Alg. 18.

A sweep is defined by its direction $\mathbf{d}_s$ and its aperture a. A sweep can be seen as solving a large linear system using Gaussian elimination, in which some unknowns are solved first and their values are used to solve the later ones, in an ordered manner. The order is given by the direction of the sweep, and the dependence of the unknowns is given by the aperture. Large apertures imply obtuse extrapolation cones, which imply the need for wide broken lines to ensure that their extrapolation cones will contain the triangle to update. On the other hand, small apertures are equivalent to acute extrapolation cones; this equivalence reduces the width of the broken line to ensure an accurate approximation.

Algorithm 17. *Sampling (Triangle to Edge)*

```
1: procedure SAMPLING($\mathcal{T}_j, \mathcal{E}_j^i$)
2:     $\mathbf{g_j} \leftarrow \mathcal{T}_j$                                   ▷ extracting the local Herglotz density
3:     $(\mathbf{H}_{i,j}) = \frac{1}{\Delta\theta} e^{ik(x_i \sin\theta_j + y_i \cos\theta_j)}$                  ▷ $(x_j, y_j) \in \mathcal{E}_j^i$
4:     $\mathbf{u} = \mathbf{Hg}$                                              ▷ sampling
5:     return $\mathbf{u}$
6: end procedure
```

Algorithm 18. *Extrapolation .*

```
1: procedure EXTRAPOLATE($\mathcal{T}_j, \{\mathcal{E}_{k(i)}^i\}_{i=0}^{N_i}$)
2:     for $1 < i \leq N_i$ do                                ▷ we have the answer if r is 0
3:         $\mathbf{u_i}$ = Sampling($\mathcal{T}_i, \mathcal{E}_{k(i)}^i$)
4:     end for
5:     $\mathbf{g_j}$ = Density Estimation($\{\mathcal{E}_{k(i)}^i\}_{i=0}^{N_i}, \{\mathbf{u_i}\}_{i=0}^{N_i}$)
6:     $\mathcal{T}_j \leftarrow \mathbf{g_j}$                                 ▷ update of the local Herglotz density
7: end procedure
```

The first step of a sweep consists in building a buffer layer of triangles at the starting boundary and update their Herglotz density using the boundary data. These triangles must have an edge on the boundary $\partial\Omega$ with an interior normal in the same direction as $\mathbf{d}_s$. Given that the boundary conditions are known, we can use Alg. 16 to update the Herglotz densities of the triangles in the buffer layer. This layer of updated triangles is used as a starting condition for the sweep. If, for example, the buffered triangles already have a non-zero density, we sample the local density on the boundary and we impose the boundary conditions using linearity.

Once the buffer layer of triangles has been updated, we proceed with the inner loop of the sweep, which can be summarized in the following steps:

1. Choose the triangle that has not been updated with the smallest center of gravity with respect to the direction of sweeping, $\mathcal{T}_j'$;

2. Choose a broken line among the updated edges such that its extrapolation cone includes $\mathcal{T}_j'$, $\{\mathcal{E}_{k(i)}^i\}_{i=0}^{N_i}$;

3. Update the local Herglotz density using Alg. 18 with $\mathbf{u_{j'}} = \texttt{Extrapolate}(\mathcal{T}_j', \{\mathcal{E}_{k(i)}^i\}_{i=0}^{N_i})$; and

4. If there exist an outdated triangle, go back to step 1, otherwise, select another direction and aperture and perform another sweep.

In practice, step 2 is precomputed. As the conditions are only geometric, a subroutine is used to set up the order in which the triangles will be visited, as well as the dependency between them. This information can be written as a directed graph. Therefore, each sweep can be easily parallelizable; a routine can be used to split the directed graph in several sub-graphs, minimizing communication.

The implementation of absorbing boundary conditions is trivial, the algorithm needs to stop at the boundary. In order to improve accuracy, we use a single extra layer of triangles to constraint the wave to escape the domain. We improve stability defining directional filters, in order to separate separate the out-going and in-going sections of the approximated Herglotz densities. The in-going part of the density is likely to be numerical noise, so filtering it out improves accuracy by improving stability when the system is inverted. We show an example of this procedure for a simple case. Fig. 4-8 depicts a Gaussian beam, which is reflected from the walls twice. The wave number is $\kappa = 50\pi$, in other words, we are able to keep an accuracy of 9 digits up to 2000 wavelengths away from the data.

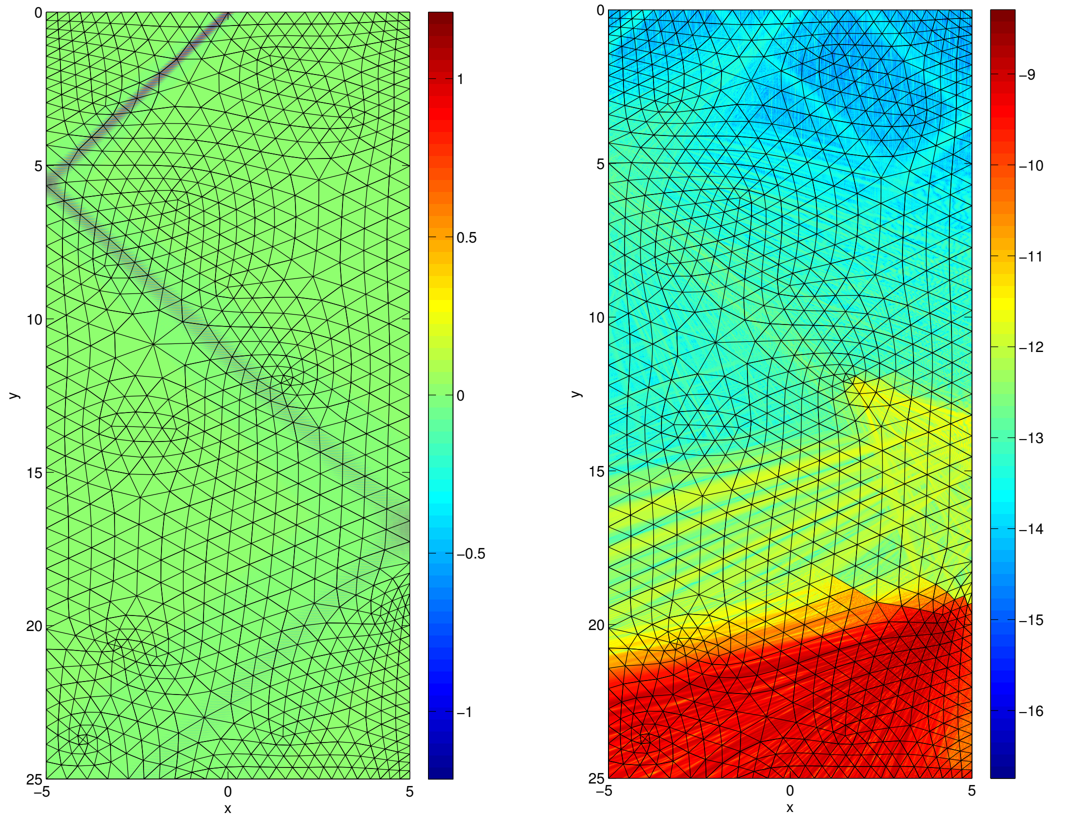


Figure 4-8: Solution of the Helmholtz equation for a Gaussian beam, with homogeneous Dirichlet boundary conditions at the left and right boundary and absorbing boundary condition at the bottom, for $\kappa = 200\pi$; *left*: real part of the solution wavefield; *right*: error between real and computed solution.

Appendix

4.A Proofs

Proof of Theorem 3

Proof. Introduce multi-indices α of length at most s, and let $M = \sum_{|\alpha|\leq s} 1$. The space $W^{-s,p}$ is identified with the dual space $(W^{s,p'})'$, and endowed with the corresponding induced norm. Here $\frac{1}{p} + \frac{1}{p'} = 1$. As a consequence of the Hahn-Banach theorem with underlying space $W^{s,p'}$, there exists a unique norm-preserving representation of $h \in (W^{s,p'})'$ as a vector-valued function $(h_\alpha) \in L^p(\mathbb{R}^M)$, i.e.,

$$\langle h, g\rangle = \sum_{|\alpha|\leq s} \int_{-1}^{1} h_\alpha \partial^\alpha g, \qquad \|h\|_{W^{-s,p}} = \|(h_\alpha)\|_{L^p(\mathbb{R}^M)}.$$

As a result,

$$u(x,y) = \sum_{|\alpha|\leq s} \int_{-1}^{1} \partial_t^\alpha \left[\exp ic \left(xt + y\sqrt{\frac{\kappa^2}{c^2} - t^2} \right) \right] h_\alpha(t)\, dt.$$

The successive derivatives bring down factors that can easily been seen to be bounded uniformly in $t \in (-1,1)$ by $C_s \left(1 + |x| + \frac{y}{\sqrt{\kappa^2/c^2 - 1}}\right)^s$ for some constant C_s. The proof is completed by pulling a factor $\|h_\alpha\|_p$ out of each integral. The sum over α only increases the overall constant $C_{p,s}$.

□

Proof of Lemma 9

Proof. Let $v \in H^s([-1,1])$. Given that the PSWF form an orthonormal basis in $L^2([-1,1])$ we have that for any $v \in L^2[-1,2]$,

$$v(x) = \sum_{l=1}^{\infty} \langle v, \psi_\ell^c\rangle \psi_\ell^c(x), \qquad \langle v, \psi_\ell^c\rangle = \int_{-1}^{1} v(t)\psi_\ell^c(t) dt.$$

Therefore, following Thm. 3.1 of [34]

$$|\langle v, \psi_\ell^c \rangle| \leq C \left(\ell^{-\frac{2}{3}s} \|v\|_{H^s([-1,1))} + q_\ell^{\delta \ell} \|v\|_{L^2([-1,1])} \right), \tag{4.39}$$

where $C > 0$ and $\delta > 0$ are constants independent of ℓ, and $q_\ell = \sqrt{\frac{c^2}{\chi_\ell^2}} < 1$ for ℓ large enough. Now pick, $v = u|_\gamma$ such that,

$$v(x) = u|_\gamma(x) = H_0[h] = \int_{-1}^{1} e^{ictx} h(t)dt.$$

Then $\langle v, \psi_\ell^c \rangle$ can be rewritten as

$$\begin{aligned}
\langle v, \psi_\ell^c \rangle = \int_{-1}^{1} v(x)\psi_\ell^c(x) &= \int_{-1}^{1} \left[\int_{-1}^{1} e^{ictx} h(t)dt \right] \psi_\ell^c(x)dx & (4.40)\\
&= \int_{-1}^{1} \left[\int_{-1}^{1} e^{ictx} \psi_\ell^c(x)dx \right] h(t)dt & (4.41)\\
&= \int_{-1}^{1} \lambda_\ell^c \psi_\ell^c(t)h(t)dt & (4.42)\\
&= \lambda_\ell^c \langle h, \psi_\ell^c \rangle. & (4.43)
\end{aligned}$$

By assumption, $g \in H^s[-1,1]$, thus by combining Eq. 4.39 and Eq. 4.43 we have

$$|\langle v, \psi_\ell^c \rangle| \leq C|\lambda_\ell| \left(\ell^{-\frac{2}{3}s} \|h\|_{H^s([-1,1))} + q_\ell^{\delta \ell} \|h\|_{L^2([-1,1])} \right), \tag{4.44}$$

which has a geometrically faster decay than the decay of the eigenvalues.

Given that we picked $v = u|_\gamma$ and following the definition of the truncated inverse we have that

$$\|h(t) - (H_{0,N}^{-1} u|_\gamma)(t)\|_{L^2} \leq \sum_{k=N+1}^{\infty} \left| \frac{\langle u|_\gamma, \psi_k^c \rangle}{\lambda_k} \right| \|\psi_k^c\| \quad = \quad \sum_{k=N+1}^{\infty} |\langle h, \psi_k^c \rangle| \,.$$

Moreover, using the bound given by Eq. 4.44 we obtain that

$$\|h(t) - (H_{0,N}^{-1} u|_\gamma)(t)\|_{L^2([-1,1])} \leq \sum_{k=N+1}^{\infty} C \left(k^{-\frac{2}{3}s} \|h\|_{H^s([-1,1))} + q_k^{\delta k} \|h\|_{L^2([-1,1])} \right). \tag{4.45}$$

Thus, it is possible to recover the Herglotz density in the limit $N \to \infty$. The precision of the truncation for a fixed N is given by the regularity of the Herglotz density. □

Proof of Lemma 10

Proof. 1. This in Thm. 14 in [106].

2. By definition,

$$\psi_n^c(x) = \int_{-1}^{1} e^{icxt} \frac{\psi_n^c(t)}{\lambda_n^c} dt,$$

and integrating by parts, we have that

$$|\psi_n^c(x)| \leq \frac{|\psi_n^c(-1)| + |\psi_n^c(1)|}{|cx|} - \frac{1}{|\lambda_n^c cx|} \int_{-1}^{1} \left| \frac{d}{dt} \psi_n^c(t) \right| dt,$$

which implies $\psi_n^c(x) \sim \frac{1}{|x|}$ as $|x| \to \infty$.

3. Since ψ_n^c is continuous and vanishing at infinity we can form a sequence $\{z_n\}_{n=1}^{\infty}$ where z_n is such that $\max_{x\in\mathbb{R}} |\lambda_n^c \psi_n^c(x)| = |\lambda_n^c \psi_n^c(z_n)|$. For each n, z_n may not be unique but it always exits and is finite by the by the argument above.

We define $\varphi_n(x) = (\lambda_n^c \psi_n^c \eta_2)(x - z_n)$, where $\eta_2 \in \mathcal{C}_c^{\infty}(\mathbb{R})$ a smooth cut-off such that

$$\eta_2(x) = \begin{cases} 1 & \text{if } x < 1 \\ 0 & \text{if } x > 2 \end{cases}. \tag{4.46}$$

By definition, φ_n is supported on $[-2, 2]$ and continuous, hence we can write

$$|\varphi_n(x)| = \left| \int_{-2}^{x} \varphi_n'(y) dy \right| \leq \int_{-2}^{2} |\varphi_n'(y)| dy \leq 2|\varphi_n|_{H^1(-2,2)} = 2|\varphi_n|_{H^1(\mathbb{R})}. \tag{4.47}$$

Moreover, using the Bernstein inequality on $\lambda_n^c \psi_n^c$ we have that

$$|\varphi|_{H^1(\mathbb{R})} = \left(\int_{\mathbb{R}} |\varphi'|^2 \right)^{1/2} = \left(\int_{\mathbb{R}} \left| \frac{d\eta}{dy} \lambda_n^c \psi_n^c + \eta \lambda_n^c \frac{\psi_n^c}{dy} \right|^2 \right)^{1/2} \leq \widetilde{C} c \|\lambda_n^c \psi_n^c\|_{L^2(\mathbb{R})} \leq \widetilde{C} c. \tag{4.48}$$

Combining Eq. 4.48 and Eq. 4.47 and taking the maximum, we have

$$\|\lambda_n^c \psi_n^c\|_{L^\infty(\mathbb{R})} = \|\varphi_n\|_{L^\infty(\mathbb{R})} \leq \widetilde{C} c. \tag{4.49}$$

which concludes the proof.

□

Proof of Lemma 11

Proof. Let $\kappa/c > 2$ and fix $yc^2 > \sqrt{\kappa^2 - c^2}$. For clarity let

$$f(t) = e^{iy\sqrt{\kappa^2 - c^2 t^2}}, \qquad \phi(t) = \sqrt{\kappa^2 - c^2 t^2}.$$

Let fix α in a small neighborhood of 1, the proof of this Lemma is based on successive expansions of the derivatives, in addition to the fact that α lies in a small neighborhood of 1. In that regime, the derivatives of $\sqrt{\kappa^2 - c^2 t^2}$ in $1 < |t| < \alpha$ are controlled by the derivatives at $t = 1$ times an amplification factor in the form of a constant. Using the Leibniz formula, the derivatives of $e^{iy\sqrt{\kappa^2-c^2t^2}}\eta_\alpha(t)$ are

$$\partial_t^n\left(e^{icy\sqrt{(\kappa/c)^2-t^2}}\eta_\alpha(t)\right)=\sum_{k=0}^{n}\binom{n}{k}\partial_t^k f(t)\partial_t^{n-k}\eta_\alpha(t).$$

This formula can be simplified in some cases, for example, if $|t|<1$, thus η_α is constant and all of its derivatives vanish. However, if $1<|t|<\alpha$, we need to analyze the derivatives of each function and, in particular, consider the leading terms, i.e.,

$$\begin{aligned}\partial_t^n f(t) &= \partial_t^n\left(e^{iy\phi(t)}\right), &(4.50)\\ &= \sum\frac{n!}{m_1!m_2!...m_n!}(iy)^{\sum_{j=1}^n m_j}e^{iy\phi(t)}\prod_{j=1}^{n}\left(\frac{\phi^{(j)}}{j!}\right)^{m_j}, &(4.51)\end{aligned}$$

with the constraint that $\sum_{j=1}^n jm_j=n$. We want to prove that the leading term in Eq. 4.51 is the monomial. In order to obtain this intermediate result, we need to control the other coefficients of the right-hand side of Eq. 4.51. To obtain the necessarily bound we compute the higher order derivatives of ϕ using the Faà di Bruno's formula resulting in,

$$\phi^{(n)}(s)=\sum\frac{n!}{m_1!m_2!...m_n!}\left(\prod_{k=1}^{\sum m_j}\frac{-2k+3}{2}\right)\left(\kappa^2-c^2t^2\right)^{-\sum m_j+1/2}\prod_{j=1}^{2}\left(\frac{-(c^2t^2)^{(j)}}{j!}\right)^{m_j}, \tag{4.52}$$

with the constraint that $\sum_j m_j j=n$. We observe that $(t^2)^{(j)}=0\ \forall j>2$, which simplifies the constraint to $m_1+2m_2=n$ and henceforth Eq. 4.52 to

$$\phi^{(n)}(t)=\frac{c^2}{\sqrt{\kappa^2-c^2t^2}}\left(\sum_{m_1,m_2}C_{n,m_1}\left(\frac{c}{\sqrt{\kappa^2-c^2t^2}}\right)^{2(m_1+m_2-1)}t^{m_1}\right), \tag{4.53}$$

which can be bounded for $t\in[-\alpha,\alpha]$ by

$$|\phi^{(n)}(t)|\le C_{\alpha,n}\frac{c^2}{\sqrt{\kappa^2-c^2}}\left(\frac{c}{\sqrt{\kappa^2-c^2}}\right)^{n-1}, \tag{4.54}$$

where we used the hypothesis $2c<\kappa$ that implies that $c/\sqrt{\kappa^2-c^2}<1/\sqrt{3}$, and the fact that α is in a neighborhood of 1. Moreover, Eq. 4.54 can be used to bound $\partial_t^n f(t)$ in Eq. 4.51 obtaining that

$$\begin{aligned}|\partial_t^n f(t)| &\le \sum C_{\alpha,n}\left(\frac{c^2y}{\sqrt{\kappa^2-c^2}}\right)^{\sum_{j=1}^n m_j}\prod_{j=1}^{n}\left(\left(\frac{c}{\sqrt{\kappa^2-c^2}}\right)^{j-1}\right)^{m_j}, &(4.55)\\ &\le \sum C_{\alpha,n}\left(\frac{c^2y}{\sqrt{\kappa^2-c^2}}\right)^{\sum_{j=1}^n m_j}\left(\frac{c}{\sqrt{\kappa^2-c^2}}\right)^{n-\sum_{j=1}^n m_j}. &(4.56)\end{aligned}$$

Finally, given that $c/\sqrt{\kappa^2 - c^2} < 1/\sqrt{3}$, $c^2 y/\sqrt{\kappa^2 - c^2} > 1$ and $n - \sum_{j=1}^n m_j > 0$ unless $m_1 = n$, we obtain than

$$|\partial_t^n f(t)| \leq C_{n,\alpha} \left(\frac{c^2 y}{\sqrt{\kappa^2 - c^2}} \right)^n . \tag{4.57}$$

If $c^2 y < \sqrt{\kappa^2 - c^2}$ a slight modification of the argument showed above can be used to prove that $|\partial_t^n f(t)| < C_{\alpha,n}$

The function $\eta_\alpha(t)$ is independent of k, c and y, hence its derivatives are $\mathcal{O}(1)$ and

$$|\partial_t^n \eta_\alpha(t)| \leq C_{n,\alpha.} \tag{4.58}$$

Substituting Eq. 4.57 and Eq. 4.58 in the Leibniz formula, we obtain the bound

$$|\partial_t^n \left(e^{icy\sqrt{(\kappa/c)^2 - t^2}} \eta_\alpha(t) \right) | \leq \sum \binom{n}{k} C_{k,\alpha} \left(\frac{c^2 y}{\sqrt{\kappa^2 - c^2}} \right)^k . \tag{4.59}$$

Therefore, keeping the leading term and using that $c^2 y/\sqrt{\kappa^2 - c^2} > 1$, the last bound can be simplified resulting in in

$$|\partial_t^n \left(e^{icy\sqrt{(\kappa/c)^2 - t^2}} \eta_\alpha(t) \right) | \leq C_{n,\alpha} \left(\frac{c^2 y}{\sqrt{\kappa^2 - c^2}} \right)^n , \tag{4.60}$$

which can be slightly modified when $c^2 y < \sqrt{\kappa^2 - c^2}$ to yield the desired result. $\square$

Proof of Lemma 12 . Define the self-adjoint operator :

$$\mathcal{L} = \frac{(I - \frac{1}{c_y^2} \triangle_t)}{1 + \frac{x^2}{c_y^2}},$$

such that $\mathcal{L} e^{ixt} = e^{ixt}$. This implies that

$$K_y(x) = \int_{-\kappa/c}^{\kappa/c} \mathcal{L}^m \left(e^{ixt} \right) e^{icy\sqrt{(\kappa/c)^2 - t^2}} \eta_\alpha(t) dt.$$

Using integration by parts, the convolution kernel can be rewritten as

$$K_y(x) = \int_{-\kappa/c}^{\kappa/c} e^{ixt} \mathcal{L}^m \left(e^{icy\sqrt{(\kappa/c)^2 - t^2}} \eta_\alpha(t) \right) dt,$$

because, by construction, $\partial_t^m \eta_\alpha(k/c) = 0$ for all $m \in \mathbb{N}$. Hence,

$$|K_y(x)| \leq C_m \left(1 + \frac{x^2}{c_y^2} \right)^{-m} \int_{-\kappa/c}^{\kappa/c} \left(I - \frac{\triangle}{c_y^2} \right)^m \left(e^{icy\sqrt{(\kappa/c)^2 - t^2}} \eta_\alpha(t) \right) dt.$$

In addition, using the bound in Eq. 4.23 we have that

$$\left| \left(I - \frac{\triangle}{c_y^2} \right)^m \left(e^{icy\sqrt{(\kappa/c)^2 - t^2}} \eta_\alpha(t) \right) \right| \leq C_{m,\alpha},$$

which leads to

$$|K_y(x)| \leq C_{m,\alpha} \left(1 + \frac{x^2}{c_y^2} \right)^{-m} \leq C_{m,\alpha} \left(1 + \frac{|x|}{c_y} \right)^{-2m}, \tag{4.61}$$

which concludes the proof. The argument for odd n is essentially the same. $\square$

Bibliography

[1] S. Ambikasaran and E. Darve. An $\mathcal{O}(n \log n)$ fast direct solver for partial hierarchically semi-separable matrices. *Journal of Scientific Computing*, 57(3):477–501, December 2013.

[2] A. Aminfar, S. Ambikasaram, and E. Darve. A fast block low-rank dense solver with applications to finite-element matrices. *ArXiv e-prints*, [cs.NA] arXiv:1403.5337, 2015.

[3] A. Aminfar and E. Darve. A fast sparse solver for finite-element matrices. *ArXiv e-prints*, [cs.NA] arXiv:1410.2697, 2014.

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

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

[6] G. Ballard, J. Demmel, O. Holtz, and O. Schwartz. Minimizing communication in numerical linear algebra. *SIAM Journal on Matrix Analysis and Applications*, 32(3):866–901, 2011.

[7] A. H. Barnett and T. Betcke. An exponentially convergent nonpolynomial finite element method for time-harmonic scattering from polygons. *SIAM Journal on Scientific Computing*, 32(3):1417–1441, 2010.

[8] A. H. Barnett, B. J. Nelson, and J. M. Mahoney. High-order boundary integral equation solution of high frequency wave scattering from obstacles in an unbounded linearly stratified medium. *ArXiv e-prints*, 2014.

[9] 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.

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

[11] J.-D. Benamou and B. Després. A domain decomposition method for the Helmholtz equation and related optimal control problems. *Journal of Computational Physics*, 136(1):68–82, 1997.

[12] M. Benzi, J. C. Haws, and M. Tuma. Preconditioning highly indefinite and nonsymmetric matrices. *SIAM Journal on Scientific Computing*, 22(4):1333–1353, 2000.

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

[14] A. J. Berkhout. *Seismic Migration: Imaging of Acoustic Energy by Wave Field Extrapolation.* Elsevier, 1980.

[15] T. Betcke, S. N. Chandler-Wilde, I. G. Graham, S. Langdon, and M. Lindner. A fast sparse solver for finite-element matrices. *ArXiv e-prints*, [math.NA] arXiv:1007.3074, 2010.

[16] T. Betcke and E. A. Spence. Numerical estimation of coercivity constants for boundary integral operators in acoustic scattering. *SIAM Journal on Numerical Analysis*, 49(4):1572–1601, 2011.

[17] R. Bevilacqua, B. Codenotti, and F. Romani. Parallel solution of block tridiagonal linear systems. *Linear Algebra and its Applications*, 104(0):39–57, 1988.

[18] G. Beylkin and K. Sandberg. Wave propagation using bases for bandlimited functions. *Wave Motion*, 41(3):263–291, 2005. Special Issue in Honor of the 75th Birthday of A.T.de Hoop.

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

[20] S. Boerm, L. Grasedyck, and W. Hackbusch. *Hierarchical matrices.* Max-Planck- Institute Lecture Notes, 2006.

[21] M. Bollhöfer, M. Grote, and O. Schenk. Algebraic multilevel preconditioner for the Helmholtz equation in heterogeneous media. *SIAM Journal on Scientific Computing*, 31(5):3781–3805, 2009.

[22] J. P. Boyd. Algorithm 840: Computation of grid points, quadrature weights and derivatives for spectral element methods using prolate spheroidal wave functions—prolate elements. *ACM Trans. Math. Softw.*, 31(1):149–165, March 2005.

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

[24] J. Bremer, A. Gillman, and P.-G. Martinsson. A high-order accurate accelerated direct solver for acoustic scattering from surfaces. *BIT Numerical Mathematics*, pages 1–31, 2014.

[25] O. Bruno, T. Elling, R. Paffenroth, and C. Turc. Electromagnetic integral equations requiring small numbers of Krylov-subspace iterations. *Journal of Computational Physics*, 228(17):6169–6183, September 2009.

[26] M. Byckling and M. Huhtanen. Preconditioning with direct approximate factoring of the inverse. *SIAM Journal on Scientific Computing*, 36(1):A88–A104, 2014.

[27] 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.

[28] E. Candès, L. Demanet, and L. Ying. A fast butterfly algorithm for the computation of fourier integral operators. *Multiscale Modeling & Simulation*, 7(4):1727–1750, 2009.

[29] O. Cessenat. *Application d'une nouvelle formulation variationnelle aux équations d'ondes harmoniques*. PhD thesis, Université Paris IX Dauphine, Paris, France, 1996.

[30] O. Cessenat and B. Després. Application of an ultra weak variational formulation of elliptic pdes to the two-dimensional Helmholtz problem. *SIAM Journal on Numerical Analysis*, 35(1):255–299, 1998.

[31] 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.

[32] S. N. Chandler-Wilde. Boundary value problems for the Helmholtz equation in a half-plane. In *Proceedings of the 3rd International Conference on Mathematical and Numerical Aspects of Wave Propagation*, page 188âĂŞ197, 1995.

[33] 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.

[34] Q. Y. Chen, D. Gottlieb, and J. S. Hesthaven. Spectral methods based on prolate spheroidal wave functions for hyperbolic PDEs. *SIAM Journal on Numerical Analysis*, 43(5):1912–1933, 2005.

[35] Y. Chen. Inverse scattering via Heisenberg's uncertainty principle. *Inverse Problems*, 13(2):253, 1997.

[36] Z. Chen, D. Cheng, and T. Wu. A dispersion minimizing finite difference scheme and preconditioned solver for the 3D Helmholtz equation. *Journal of Computational Physics*, 231(24):8152–8175, 2012.

[37] 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.

[38] 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.

[39] H. Cheng, W. Y. Crutchfield, Z. Gimbutas, L. F. Greengard, J. F. Ethridge, J. Huang, V. Rokhlin, N. Yarvin, and J. Zhao. A wideband fast multipole method for the helmholtz equation in three dimensions. *Journal of Computational Physics*, 216(1):300 – 325, 2006.

[40] M. H. Cho and A. H. Barnett. Robust fast direct integral equation solver for quasi-periodic scattering problems with a large number of layers. *Opt. Express*, 23(2):1775–1799, Jan 2015.

[41] E. Chow and Y. Saad. Approximate inverse preconditioners via sparse-sparse iterations. *SIAM Journal on Scientific Computing*, 19(3):995–1023, 1998.

[42] D. Colton and R. Kress. On the denseness of herglotz wave functions and electromagnetic herglotz pairs in sobolev spaces. *Mathematical Methods in the Applied Sciences*, 24(16):1289–1303, 2001.

[43] D. Colton and R. Kress. *Integral Equation Methods in Scattering Theory.* Society for Industrial and Applied Mathematics, Philadelphia, PA, 2013.

[44] L. Conen, V. Dolean, Krause R., and F. Nataf. A coarse space for heterogeneous helmholtz problems based on the dirichlet-to-neumann operator. *Journal of Computational and Applied Mathematics*, 271(0):83 – 99, 2014.

[45] S. Cools, B. Reps, and W. Vanroose. A new level-dependent coarse grid correction scheme for indefinite Helmholtz problems. *Numerical Linear Algebra with Applications*, 21(4):513–533, 2014.

[46] 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.

[47] S. Cools and W. Vanroose. Generalization of the complex shifted Laplacian: on the class of expansion preconditioners for Helmholtz problems. *ArXiv e-prints*, 2015.

[48] M. Darbas, E. Darrigrand, and Y. Lafranche. Combining analytic preconditioner and fast multipole method for the 3-D Helmholtz equation. *Journal of Computational Physics*, 236(0):289 – 316, 2013.

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

[50] M. V. de Hoop, S. Wang, and J. Xia. On 3D modeling of seismic wave propagation via a structured parallel multifrontal direct Helmholtz solver. *Geophysical Prospecting*, 59(5):857–873, 2011.

[51] M.V. de Hoop, J. H. Le Rousseau, and R-S. Wu. Generalization of the phase-screen approximation for the scattering of acoustic waves. *Wave Motion*, 31(1):43–70, 2000.

[52] L. Demanet and L. Ying. Fast wave computation via Fourier integral operators. *Mathematics of Computation*, 81(279):1455–1486, 2012.

[53] J. Demmel, L. Grigori, M. Gu, and H. Xiang. Communication avoiding rank revealing qr factorization with column pivoting. Technical Report UCB/EECS-2013-46, EECS Department, University of California, Berkeley, May 2013.

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

[55] 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.

[56] H. Elman, O. Ernst, and D. O'Leary. A multigrid method enhanced by Krylov subspace iteration for discrete Helmholtz equations. *SIAM Journal on Scientific Computing*, 23(4):1291–1315, 2001.

[57] B. Engquist and L. Ying. Fast directional algorithms for the Helmholtz kernel. *Journal of Computational and Applied Mathematics*, 234(6):1851 – 1859, 2010. Eighth International Conference on Mathematical and Numerical Aspects of Waves (Waves 2007).

[58] 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.

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

[60] 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.

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

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

[63] 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.

[64] 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.

[65] C. Farhat, P. Avery, R. Tezaur, and J. Li. FETI-DPH: A dual-primal domain decomposition method for acoustic scattering. *Journal of Computational Acoustics*, 13(03):499–524, 2005.

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

[67] C. Farhat, A. Macedo, and M. Lesoinne. A two-level domain decomposition method for the iterative solution of high frequency exterior Helmholtz problems. *Numerische Mathematik*, 85(2):283–308, 2000.

[68] L. Fishman, M. V. de Hoop, and M. J. N. van Stralen. Exact constructions of square-root Helmholtz operator symbols: The focusing quadratic profile. *Journal of Mathematical Physics*, 41(7):4881–4938, 2000.

[69] M. Gander and F. Nataf. AILU for Helmholtz problems: a new preconditioner based on an analytic factorization. *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics*, 331(3):261–266, 2000.

[70] 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.

[71] 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.

[72] 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.

[73] 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.

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

[75] 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.

[76] 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.

[77] I.G. Graham, M. Löhndorf, J.M. Melenk, and E.A. Spence. When is the error in the h-BEM for solving the Helmholtz equation bounded independently of k? *BIT Numerical Mathematics*, 55(1):171–214, 2015.

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

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

[80] 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.

[81] A. Hoffman, M. Martin, and D. Rose. Complexity bounds for regular finite difference and finite element grids. *SIAM Journal on Numerical Analysis*, 10(2):364–369, 1973.

[82] L. Hörmander. *The Analysis of Linear Partial Differential Operators. IV: Fourier Integral Operators*, volume 63 of *Classics in Mathematics*. Springer, Berlin, 2009.

[83] F. Ihlenburg and I. Babuska. Finite element solution of the Helmholtz equation with high wave number part I: The h-version of the FEM. *Computers & Mathematics with Applications*, 30(9):9 – 37, 1995.

[84] L.-M. Imbert-Gérard. Interpolation properties of generalized plane waves. *Numerische Mathematik*, pages 1–29, 2015.

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

[86] P. Jones, J. Ma, and V. Rokhlin. A fast direct algorithm for the solution of the Laplace equation on regions with fractal boundaries. *Journal of Computational Physics*, 113(1):35–51, 1994.

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

[88] J. Lai, S. Ambikasaran, and L Greengard. A fast direct solver for high frequency scattering from a large cavity in two dimensions. *ArXiv e-prints*, 2014.

[89] H. J. Landau and H. O. Pollak. Prolate spheroidal wave functions, Fourier analysis and uncertainty - II. *Bell System Technical Journal*, 40:65–85, 1961.

[90] H. J. Landau and H. O. Pollak. Prolate spheroidal wave functions, Fourier analysis and uncertainty - III. the dimension of the space of essentially time- and band-limited signals. *Bell System Technical Journal*, 41:1295–1336, 1962.

[91] L. Lin, J. Lu, and L. Ying. Fast construction of hierarchical matrix representation from matrix-vector multiplication. *Journal of Computational Physics*, 230(10):4071–4087, May 2011.

[92] J. L. Lions and E. Magenes. *Non-Homogeneous Boundary Value Problems and Applications*, volume 1 of *Grundlehren der mathematischen Wissenschaften*. Springer-Verlag, 1972. ISBN 978-3-642-65161-8.

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

[94] 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.

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

[96] 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.

[97] 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.

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

[99] G. Meurant. A review on the inverse of symmetric tridiagonal and block tridiagonal matrices. *SIAM Journal on Matrix Analysis and Applications*, 13(3):707–728, July 1992.

[100] 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.

[101] A. Moiola, R. Hiptmair, and I. Perugia. Vekua theory for the Helmholtz operator. *Zeitschrift fÃijr angewandte Mathematik und Physik*, 62(5):779–807, 2011.

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

[103] P. Monk and D.-Q. Wang. A least-squares method for the Helmholtz equation. *Computer Methods in Applied Mechanics and Engineering*, 175(1âĂŞ2):121–136, 1999.

[104] D. Osei-Kuffuor, R. Li, and Y. Saad. Matrix reordering using multilevel graph coarsening for ILU preconditioning. *SIAM Journal on Scientific Computing*, 37(1):A391–A419, 2015.

[105] A. Osipov. Certain upper bounds on the eigenvalues associated with prolate spheroidal wave functions. *Applied and Computational Harmonic Analysis*, 35(2):309 – 340, 2013.

[106] A. Osipov and V. Rokhlin. Detailed analysis of prolate quadratures and interpolation formulas. *ArXiv e-prints*, 2012.

[107] C. S. Peskin. Numerical analysis of blood flow in the heart. *Journal of Computational Physics*, 25(3):220 – 252, 1977.

[108] R.-E. Plessix. A Helmholtz iterative solver for 3D seismic-imaging problems. *Geophysics*, 72(5):SM185–SM194, 2007.

[109] 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.

[110] S. D. Poisson. *Théorie mathématique de la chaleur*. Bachelier, Paris, 1835.

[111] J. Poulson, L. Demanet, N. Maxwell, and L. Ying. A parallel butterfly algorithm. *SIAM Journal on Scientific Computing*, 36(1):C49–C65, 2014.

[112] 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.

[113] R. G. Pratt. Seismic waveform inversion in the frequency domain; part 1: Theory and verification in a physical scale model. *Geophysics*, 64(3):888–901, 1999.

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

[115] C. D. Riyanti, A. Kononov, Y. A. Erlangga, C. Vuik, C. W. Oosterlee, R.-E. Plessix, and W. A. Mulder. A parallel multigrid-based preconditioner for the 3D heterogeneous high-frequency Helmholtz equation. *Journal of Computational Physics*, 224(1):431–448, 2007. Special Issue Dedicated to Professor Piet Wesseling on the occasion of his retirement from Delft University of Technology.

[116] V. Rokhlin. Rapid solution of integral equations of classical potential theory. *Journal of Computational Physics*, 60(2):187–207, 1985.

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

[118] K. Sandberg and G. Beylkin. Full-wave-equation depth extrapolation for migration. *Geophysics*, 74(6):WCA121–CA128, 2009.

[119] 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.

[120] M. Sini and N. T. Thành. Inverse acoustic obstacle scattering problems using multifrequency measurements. *Inverse Problems and Imaging*, 6(4):749–773, 2012.

[121] D. Slepian. Prolate spheroidal wave functions, fourier analysis and uncertainity - IV. extensions to many dimensions; generalized prolate spheroidal functions. *Bell System Technical Journal*, 43:3009–3057, 1964.

[122] D. Slepian. Some comments on fourier analysis, uncertainty and modeling. *SIAM Review*, 25(3):379–393, 1983.

[123] D. Slepian and H. O. Pollak. Prolate spheroidal wave functions, Fourier analysis and uncertainty - I. *Bell System Technical Journal*, 40:43–63, 1961.

[124] E. A. Spence. Wavenumber-explicit bounds in time-harmonic acoustic scattering. *SIAM Journal on Mathematical Analysis*, 46(4):2987–3024, 2014.

[125] E. A. Spence. Bounding acoustic layer potentials via oscillatory integral techniques. *BIT Numerical Mathematics*, 55(1):279–318, 2015.

[126] E. A. Spence, S. N. Chandler-Wilde, I. G. Graham, and V. P. Smyshlyaev. A new frequency-uniform coercive boundary integral equation for acoustic scattering. *Communications on Pure and Applied Mathematics*, 64(10):1384–1415, 2011.

[127] P. L. Stoffa, J. T. Fokkema, R. M. de Luna Freire, and W. P. Kessinger. Split-step fourier migration. *Geophysics*, 55(4):410–421, 1990.

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

[129] C. Stolk and M. de Hoop. Modeling of seismic data in the downward continuation approach. *SIAM Journal on Applied Mathematics*, 65(4):1388–1406, 2005.

[130] C. C. Stolk. A dispersion minimizing scheme for the 3-D Helmholtz equation with applications in multigrid based solvers. *ArXiv e-prints*, 2015.

[131] C. C. Stolk, M. Ahmed, and S. K. Bhowmik. A multigrid method for the Helmholtz equation with optimized coarse grid corrections. *ArXiv e-prints*, [math.NA] 1304.4103, April 2013.

[132] G. Strang and T. Nguyen. The interplay of ranks of submatrices. *SIAM Review*, 46(4):637–646, 2004.

[133] W. W. Symes and J. J. Carazzone. Velocity inversion by differential semblance optimization. *GEOPHYSICS*, 56(5):654–663, 1991.

[134] A. Tarantola. Inversion of seismic reflection data in the acoustic approximation. *Geophysics*, 49(8):1259–1266, 1984.

[135] M. E. Taylor. Grazing rays and reflection of singularities of solutions to wave equations. *Communications on Pure and Applied Mathematics*, 29(1):1–38, 1976.

[136] R. Thakur, R. Rabenseifner, and W. Gropp. Optimization of collective communication operations in mpich. *International Journal of High Performance Computing Applications*, 19(1):49–66, 2005.

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

[138] 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.

[139] 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.

[140] 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.

[141] R. A. Usmani. Inversion of Jacobi's tridiagonal matrix. *Computers & Mathematics with Applications*, 27(8):59–66, 1994.

[142] 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.

[143] 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.

[144] 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.

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

[146] H. Xiao, V. Rokhlin, and N. Yarvin. Prolate spheroidal wavefunctions, quadrature and interpolation. *Inverse Problems*, 17(4):805, 2001.

[147] P. Ylä-Oijala, M. Taskinen, and S Järvenpää. Analysis of surface integral equations in electromagnetic scattering and radiation problems. *Engineering Analysis with Boundary Elements*, 32(3):196 – 209, 2008.

[148] L. Zepeda-Núñez and L. Demanet. The method of polarized traces for the 2D Helmholtz equation. *ArXiv e-prints*, 2014.