Université Catholique de Louvain
Institute of Mechanics, Materials and Civil Engineering
Mécanique et Mathématiques Appliquées
Université de Liège
Montefiore Institute
Applied and Computational Electromagnetics

Generalized sweeping preconditioners for domain decomposition methods applied to Helmholtz problems

Ruiyang DAI

Doctoral dissertation presented
in fulfillment of the requirements for the degree of
Docteur en Sciences de l'Ingénieur

in the

Hextreme team & ACE team

Friday 1st October, 2021

Thesis Committee:

Prof. Jean-François Remacle **Promoteur**	UCLouvain, Belgium
Prof. Christophe Geuzaine **Promoteur**	Université de Liège, Belgium
Prof. Philippe Chatelain **Jury**	UCLouvain, Belgium
Ph.D Jonathan Lambrechts **Jury**	UCLouvain, Belgium
Prof. Xavier Antoine **Jury**	Université de Lorraine, France
Ph.D Eric Béchet **Jury**	Université de Liège, Belgium
Ph.D Axel Modave **Jury**	ENSTA Paris, France
Prof. Renaud Ronsse **Président**	UCLouvain, Belgium

Version history:

April 2020	- beginning date
May 2021	- first draft
Juin 2021	- major modifications
August 2021	- final version

"*Je pense qu'il devrait y avoir plus d'une voix dans une société saine.*"

Dr W. LI

THE LANCET, Volume 395, Issue 10225

Abstract

Keywords: Domain decomposition methods, Helmholtz problems, preconditioners.

The numerical solution of high-frequency Helmholtz problems by discretization methods such as the finite element method is a big challenge. Indeed, obtaining high-fidelity solutions requires to assemble and solve extremely large linear systems, whose size increases more than linearly with frequency. This can quickly lead to intractable computational costs both in terms of the assembly and the solution of the resulting linear systems. To overcome these difficulties, we implement an efficient quadrature approach applied to the high-order finite element method as well as domain decomposition methods with high-order transmission conditions, in two- and three dimensions, on high-performance computers. To improve the convergence rate of domain decomposition methods, we generalize a family of sweeping preconditioners for the domain decomposition methods, where sweeps can be done in several directions on block-type partitions. In order to apply our algorithms to practical cases that require solutions for large number of frequencies or a large number of right-hand sides, we also propose improved parallelization strategies that maintain the fast rate of convergence while maximizing the usage of computer resources.

Résumé

Mots clés: Méthodes de décomposition de domaine, problèmes de Helmholtz, préconditionneurs.

La résolution numérique de problèmes de Helmholtz à haute fréquence par des méthodes de discrétisation telles que la méthode des éléments finis constitute un énorme défi. En effet, l'obtention de solutions de haute fidélité nécessite d'assembler et de résoudre des systèmes linéaires extrêmement grands, dont la taille augmente plus que linéairement avec la fréquence. Ceci conduit rapidement à des coûts de calcul prohibitifs tant au niveau de l'assemblage que de la résolution des systèmes linéaires résultants. Pour surmonter ces difficultés, nous mettons en oeuvre une approche efficace de quadrature appliquée à la méthode des éléments finis d'ordre élevé, ainsi que des méthodes de décomposition de domaine avec des conditions de transmission d'ordre élevé, en deux et trois dimensions, sur des architectures de calcul haute performance. Pour améliorer le taux de convergence des méthodes de décomposition de domaine, nous généralisons une famille de préconditionneurs à balayage pour les méthodes de décomposition de domaine, où les balayages peuvent être effectués dans plusieurs directions sur des partitions cartésiennes. Afin d'appliquer nos algorithmes à des cas pratiques qui nécessitent des solutions pour un grand nombre de fréquences ou un grand nombre de membres de droite, nous proposons également des stratégies de parallélisation améliorées qui maintiennent le taux de convergence rapide tout en maximisant l'utilisation des ressources informatiques.

Remerciements

En 17 juillet 2016, un garçon qui travaillait à Nantes avec Nicolas Chevaugeon et eut introduit par lui prit l'avion à Louvain-la-Neuve, en Belgique, pour rendre visite à son promoteur Jean-François Remacle. C'était la première fois qu'il visitait ce petit pays proche de la France mais, pour lui, étranger. Il marcha un long d'une route qu'il sentait accidentée [1], depuis la gare de Louvain-la-Neuve jusqu'au bout, où se trouve le bâtiment Euler. Après la marche difficile, il rencontra son promoteur et il fut son première impression de lui était passionnée et décontractée. Son histoire commença par ce jour.

J-F reconnut sa qualité sans hésitation et l'accorda de faire un doctorat sous son aile après un entretien agréable. J-F lui donna beaucoup de confiance et liberté dans sa recherche, même si J-F ne fut jamais l'air vraiment sérieux. L'histoire ne s'eut pas limité à Louvain-la-Neuve. J-F l'emmena à l'institut Montefiore à Liège pour rencontrer son deuxième promoteur, Christhophe Geuzaine. Le jour où il rencontra Christophe, il le présenta à tout le monde, ce qui ouvrit un nouveau chapitre sur un voyage entre Louvain-la-Neuve et Liège. Chaque mois, le garçon alla à Liège une ou deux fois en voiture sans guide via l'autoroute E411 et E42 pour visiter Christophe, ce qui lui plaisait beaucoup. Christophe était très patient et attentif à répondre à ses questions, aussi simples aient été elles. Ce sont deux promoteurs, J-F et Christophe, qui enverrèrent leur enthousiasme et leur intérêt pour tous ces moments. Ce garçon ne fut jamais imaginé qu'il avait un voyage magnifique à travers le région Francophone en Belgique.

Il y avait eu beaucoup de moments heureux avec ses collègues dans leur laboratoire. Celestin, qui travaillait dans le même bureau, était un garçon intéressant. Il chanta bien, il rira avec charme, il regarda le résume de NBA souvent et le garçon l'entendit en face, et il fut beaucoup d'énergie comme son ordi bruyant qui toujours calculait intensivement sur des maillages incroyables. L'énergique du bureau d'en face, Pierre-Alexandre, qui était un garçon sociable et avait beaucoup de bavardage intéressant, était une personne tous les jours que le garçon attentait pour aller boire ensemble au Dude. Ils avaient beaucoup d'humeur et toujours lâchèrent une bonne blague, ce qui rendait le laboratoire mort plus vivant. L'entousiaste, Jovana, une belle fille

[1] Pourquoi accidentée? C'est comme mettre un bébé devant la porte de sa maison et le faire pleurer devant le monde inconnu qui l'entoure.

aux cheveux rouges du bureau d'en diagonal, qui leur dit toujours bonjour avec un sourire chaleureux et les donna des gourmandises. C'était un laboratoire merveilleux dans lequel le garçon travaillait pendant ces années. C'était un laboratoire calme et bon pour la recherche, mais aussi vivant avec beaucoup de joie pendant chaque matin, midi, et après-midi.

Il y avait eu une histoire internationale. Le garçon alla en France une fois par mois pour visiter sa copine, Yang, qui le fut toujours soutenu, partagea la vie et le supporta tous les jours. Personne ne pensait que c'était une bonne idée de maintenir une relation à distance. Personne ne pensait que c'était le bon chemin et la bonne destination pour cette relation à distance. Enfin, c'était le chemin inapproprié et la destination inadéquate qui les firent monter dans ce train fou du temps, c'était le bon compagnon et la bonne copine qui les firent monter dans ce train du temps fou. C'était un voyage inattendu!

.....

C'est une longue histoire de Ruiyang DAI en Belgique, qui ferait un livre s'il devait raconter tous les contes. Dans son livre, il y a beaucoup de personnes qu'il veut remercier. Il tient à remercier ses collègues de Bureau : Ange et sa musique énergique, Celestin et son aide indispensable pour déboguer ses codes. Il remercie P-A, Jeanne Pellerin, avec qui il a partagé des bons moments lors des conférences est des promenades à New-York. Il remercie également ses autres collègues : Kilian, Jovana, Christos, Ruili, Alexandre, Amaury, François, Jonathan, Maxence, Arthur. Il veut encore jouer aux cartes rikiki, belote avec vous. Il remercie encore ses collègues à côté de Liège: Véronique, David Colignon, Orian, Maxime, Mattéo, Anthony, Xavier, Julien, Martin, Marchner, David Gasperini.

Spécialement, il remercie les membres du jury pour leur attention, leurs commentaires, et leurs patiences sur sa thèse car il a fait une correction majeure après la défense privée. Ici, il cite : Philippe Chatelain, Jonathan Lambrechts, Xavier Antoine, Eric Béchet, Axel Modave, et Renaud Ronsse. Il remercie particulièrement Axel, avec qui il a écrit son premier papier. Axel l'a accueilli à l'ENSTA et lui a donné beaucoup de propositions et guides sur ses recherches, aussi l'a aidé pour sa répétition avec beaucoup de patience.

Il remercie également ses amis chinois qui l'accompagnaient pendant ces année à Louvain-la-Neuve. Grace à eux, sa vie a été enrichie. Ses amis dans le train de sa vie lui laissent une impression durable. Malheureusement, tout le monde doit prendre son prochain train et go on and go on. La seule chose qu'il puisse faire est d'essayer d'en tirer le meilleur parti du voyage de sa vie ..... même si le départ est tellement triste. Ici, il cite : Yiyi, Xu, Xinxin, Wenyuan, Shiwen, Junfei, Michael, Woody, Junjie, Jiaqi, Mingxi .....

Enfin, il remercie son père et sa mère pour l'avoir accompagné tout au long de ses 22 premières années de vie et de l'avoir soutenu pour passer de l'autre côté de la Terre. Il remercie son grand-père et grand-mère pour leur affection.

Cinq ans plus tard, alors qu'il fait face à la route [2], il se souvient de ce lointain après-midi mais la route n'est plus étrangère ni accidentée. Il connaît toutes les rues à Louvain-la-Neuve, il connaît tous les bâtiments à Louvain-la-Neuve, et il connaît toutes les beautés naturelles des environs de Louvain-la-Neuve, le Bois des rêves, le Golf de Louvain-la-Neuve, Céroux, Fontenelle, Tienne Verlaine et Bois Henri, etc. Comme il souhaitait pouvoir abandonner tous ses mémoires et revenir à ce lointain après-midi et parcourir cette route accidentée une fois de plus.

Ruiyang DAI

[2] Avenue Georeges Lemaître.

Acknowledgments

This work was carried out in the framework of the project “Large Scale Simulation of Waves in Complex Media” funded by the Communauté Française de Belgique under contract ARC WAVES 15/19-03.

Contents

Introduction

Wave propagation is a subject of great importance in physics. You might have read Lord Rayleigh's great book, *The Theory of Sound*, in which he deduces his scattering theory that "the ratio of the scattered and direct waves is in general proportional to the inverse square of the wave-length"[86] which explains why the sky is blue. Or you might have known from one of the top ten beautiful experiments, Newton's decomposition of sunlight, which produced a stretched image of the white sunlight diffracted by a wedge, featured a blue upper edge and red lower edge. There is a great deal of theory and experiment, for centuries, that have allowed us to better understand wave propagation in this world. Unfortunately, many wave propagation problems are too sophisticated for an analytical solution and for which an experiment could not be carried out for various reasons. Since the mid 20th century, the achievements of computer science have brought solutions for theoretical and experimental analysis, which offered the possibilities to tackle the difficulties in wave propagation problems. Therefore, computer simulations have garnered a lot of interest from researchers. In this thesis, we focus particularly on numerical tools to solve wave propagation problems.

Wave equation

The present thesis starts with a classical mathematical model: the wave equation. The wave equation is a second-order hyperbolic partial differential equation (PDE) for the mathematical description of waves, such as acoustic waves (e.g. noise from a jet engine, pressure waves in the ground following an earthquake, ultrasound in medical imaging) or light waves. It arises in fields like acoustics, electromagnetics and fluid dynamics, and has a large number of practical applications. In this thesis, the scalar cases are the major concern. Consider the propagation of waves in a domain $\Omega \subset \mathbb{R}^3$ with a local speed $c(\boldsymbol{x})$. They can be described by real-valued functions $U(\boldsymbol{x}, t)$, $\boldsymbol{x} \in \Omega$, $t \in [0, \infty]$, satisfying the scalar wave equation [55]

$$c(\boldsymbol{x})\nabla^2 U(\boldsymbol{x}, t) - \frac{\partial^2 U(\boldsymbol{x}, t)}{\partial t^2} = 0.$$

Generally, a solution of the wave equation describes a wave by a series of measurements of variations (e.g. variations in sound pressure level, etc.), over

time. To give such a useful solution, for a PDE-based mathematical model, it is generally necessary to formulate that model as a well-posed PDE problem. For hyperbolic PDEs, the proper specification is: the time t must be open in the positive direction; initial conditions must be imposed along the time boundary for $t = 0$; a single boundary condition must be specified on the boundary of Ω. Because second order time derivatives appear in the wave equation, two initial conditions are required on both $U(\boldsymbol{x}, 0)$ and $U^{'}(\boldsymbol{x}, 0)$. There are some types of boundary conditions that are often applied to PDEs. The first one is the Dirichlet condition, in which the values of the function U are prescribed on the boundary of Ω. The second is the Neumann condition, where conditions on the first partial derivatives of the function U is given. Next are Robin conditions, where linear combinations of the function U and one of its first partial derivatives are given. It is worth noting that the domain Ω can be bounded or unbounded, simply connected or multiply connected, with different boundary conditions considered on different parts of the boundary. Studying the wave equation with various conditions in a certain domain for solving physical problems has a long history. The wave equation was discovered by d'Alembert in 1746 in the one-dimensional case and later by Euler in the three-dimensional case. Researchers are still devoted to this equation because its solution can be notoriously complicated, even with the same propagation speed in the whole spacial domain. Finding the solution of the wave equation in real applications is a huge challenge.

Helmholtz problems

Assuming that the time evolution of the solution of the wave equation is harmonic with angular dimensional frequency ω:

$$\begin{aligned} U(\boldsymbol{x}, t) &= \Re\left(u(\boldsymbol{x}) e^{-i\omega t}\right) \\ &= \Re\left(u(\boldsymbol{x})\right)\cos(\omega t) + \Im\mathfrak{m}(u(\boldsymbol{x}))\sin(\omega t) \end{aligned}$$

with $u(\boldsymbol{x})$ a complex-valued function, leads to the Helmholtz equation [55]

$$\nabla^2 u(\boldsymbol{x}) + k^2(\boldsymbol{x})\, u(\boldsymbol{x}) = 0, \quad \boldsymbol{x} \in \Omega,$$

where $k(\boldsymbol{x}) = \omega / c(\boldsymbol{x})$ $[m^{-1}]$ is the wavenumber ($k(\boldsymbol{x}) \neq 0, \forall \boldsymbol{x} \in \Omega$), which can be constant in homogeneous media, or dependent on the space in heterogeneous media. The solutions $u(\boldsymbol{x})$ have an oscillatory nature; if $\Im\mathfrak{m}(k) > 0$, they are damped. The Helmholtz equation belongs to the category of elliptic problems, which requires another proper specification of the type and number of conditions: $u(\boldsymbol{x})$ must be defined in a closed domain and a single boundary condition must be specified on the boundary of the spatial domain. Mathematical models depend on the types of boundary conditions, geometries, and velocity profiles, which lead to a variety of solutions.

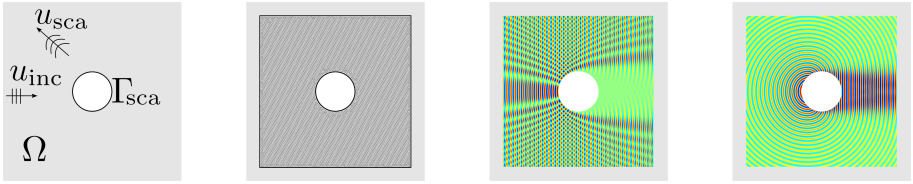


FIGURE 1: Illustration of an unbounded domain Ω and the same domain truncated by an artificial boundary, for a scattering problem by an object. The computational domain is in gray and extends to infinity in the first case, while being enclosed in an artificial boundary in the second case. The third one illustrates the total field $u_{\text{inc}} + u_{\text{scat}}$. The fourth one illustrates the scattered field u_{scat}.

It is worth noting that, for an unbounded domain Ω, the solution is not set in a closed domain, which makes the above mathematical models ill-posed. From a physical point of view, there are three types of waves in the above models: outgoing waves, ingoing waves, and standing waves. Physically we are interested in outgoing waves, i.e. waves radiating towards infinity. Ingoing waves are mathematically equivalent to waves radiating from infinity. Standing waves, which can be superimposed to any solution and still satisfy the Helmholtz equation, show the lack of uniqueness of the solution, and can be expressed as a combination of ingoing and outgoing waves. To ensure the uniqueness of the solution of the above models, a Sommerfeld radiation condition [5] is imposed at infinity, that only outgoing waves satisfy. The wave propagation models described by the Helmholtz equation in Ω and restricted by the Sommerfeld radiation condition at infinity are broadly referred to as *Helmholtz problems*.

Having established a mathematical model, one can consider a numerical model, which is the construction of an algebraic model that can be solved on a computer. We are interested to solve Helmholtz problems in the high-frequency regime (when the wavelength at the working frequency is much smaller than the size of the surrounding objects, or the computational domain), in complex geometrical domains, and in heterogeneous media, three conditions that are common to many industrial applications. For instance, in the seismic imaging industry, longitudinal wave fields in complex geological media show a wide range of space-variant wavenumbers which is caused by a wide range of velocity profiles. Another example is the simulation of the scattering of waves, such as radar or sonar imaging and wireless communications, in which the wavelength of the signal is several orders of magnitude less than the whole of the computational domain. Complex boundaries make the use of finite difference schemes sub-optimal and heterogeneity prevents the use of integral methods. Finite element methods (FEMs) [25, 1] are then the natural

choice within those hypotheses as they can handle geometrical complexity and heterogeneous media. There are some difficulties in solving Helmholtz problems using the FEM though. First, it is not trivial to handle unbounded domains. A solution is to truncate them using artificial boundaries on which appropriate boundary conditions are imposed. While the Sommerfeld radiation condition can be used on these artificial boundaries, it doesn't remain exact at a finite distance in higher-dimensional cases [44]. Several more accurate alternatives have been proposed over the years [46, 45, 56, 70, 8, 7, 44, 77]. Second, low-order FEM (e.g. with piece-wise linear shape functions) is known to be endowed with poor spectral properties and is not appropriate for high-frequency problems. Indeed, Ihlenburg and Babuška [52] evidenced that the relative error bound $e(u)$ in the H^1-seminorm of the Helmholtz equation is

$$|e|_1 \leq C_1 \left(\frac{kh}{2p}\right)^p + C_2 k \left(\frac{kh}{2p}\right)^{2p}, \tag{1}$$

where p is order of shape functions, C_1 and C_2 are constants independent of kh, and h is the size of the elements in the finite element mesh. The first term of the relative error bound reflects the error of best approximation and the second term reflects the phase difference between the exact and the Galerkin finite element solutions. This second term is known as the pollution effect (the numerical solution differs significantly from the best approximation), and is the dominating term at high wavenumber k. For wave propagation simulations, it would seem natural to keep the value kh constant (usually $kh = 0.1$). For instance, if one multiplies k by 2, one divides h by 2 to keep kh constant. However, the pollution effect prevents one to do so, as for conserving the pollution error constant, one gets a ratio R between h_k (the mesh sizes for a wavenumber k) and h_{2k} (the mesh size of a wavenumber $2k$) of

$$R = h_k / h_{2k} = 2^{(2p+1)/2p}.$$

For $p = 1$, $R \approx 2.82$ and one has to reduce the mesh size by 2.82 to have the same pollution error. A solution to reduce the pollution effect is thus clearly to increase the order of the shape functions: for $p = 7$, $R \approx 2.10$ and one have to reduce the mesh size by 2.10 to have the same pollution error. In summary, high-order FEM makes it possible to mitigate the pollution effect of low-order methods, at the price of increasing problem size faster than the wavenumber k. At high wavenumbers this leads to intractable computational costs for direct solvers. Moreover, with high-order polynomials, the assembly of the resulting linear system is also troublesome. As evidenced in[61, 68], there exists a threshold polynomial order above which the assembly time of the system starts to overshadow the computational cost of solving it. An efficient implementation can help increasing this threshold, as we will show in Chapter 1.

The other main challenge faced by many researchers who solve Helmholtz problems is the indefinite nature of the matrices resulting from standard [72, 20] discretizations. For the sake of simplicity, the Helmholtz equation is considered in a bounded domain, with zero Dirichlet boundary conditions imposed on the boundary of the domain. The discretization is written in general as

$$k^2(\boldsymbol{x})\boldsymbol{K}\boldsymbol{u} + \boldsymbol{M}\boldsymbol{u} = \boldsymbol{0}.$$

In this formulation, $\boldsymbol{u}(\boldsymbol{x})$ is the vector of discrete unknowns, $\boldsymbol{K}$ is the discrete version of the ∇^2 operator, and $\boldsymbol{M}$ is the mass matrix. Assuming a domain of size L $[m]$, the dimensionless quantity

$$H = \frac{\omega L}{c(\boldsymbol{x})} = k(\boldsymbol{x})L$$

is called the adimensional Helmholtz wave number. It represents the ratio between the wavelength of the signal $2\pi/k$ and the size L of the domain. It counts how many periods are required for a wave to cross a domain of size L. The high-frequency regime is a regime when H is large, typically $H \simeq 100$. Introducing the Helmholtz number into the discretization of the Helmholtz equation leads to the following discrete problem

$$\left(\boldsymbol{I} + \frac{L^2}{H^2}\boldsymbol{M}^{-1}\boldsymbol{K}\right)\boldsymbol{u} = \boldsymbol{0}.$$

Eigenvalues of $\boldsymbol{M}^{-1}\boldsymbol{K}$ are all less than or equal to zero. Any grid-based numerical method that aims at solving the Helmholtz equation requires a certain number N of grid points per wavelength. Note that N may depend on H and increases with the frequency. Thus, solving Helmholtz problem in the high frequency regime requires a number of grid points that is proportional to $n = (NH)^3$. Assuming for example $H = 50$ and $N = 6$, which is a good value for finite elements, about 27 million grid points are required to accurately resolve such a configuration. The smallest eigenvalue λ_1 of $\boldsymbol{M}^{-1}\boldsymbol{K}$ scales like $\lambda_1 = O(-L^{-2})$ which correspond to a resolved mode with a wavelength of the order of the domain size L. The largest eigenvalue of λ_n of $\boldsymbol{M}^{-1}\boldsymbol{K}$ scales like $\lambda_n = O(-N^2H^2L^{-2})$ which correspond to a wavelength of the order of the grid spacing $L/(NH)$. Thus eigenvalues μ of $\boldsymbol{I} + (L^2/H^2)\boldsymbol{M}^{-1}\boldsymbol{K}$ are in the range

$$1 - N^2 \leq \mu \leq 1 - \frac{1}{H^2}. \tag{2}$$

Preconditioned iterative methods are usually chosen for solving linear systems with tens of millions of degrees of freedom. Yet, (2) shows that matrices arising from the discretization of Helmholtz equations are indefinite, meaning that the spectrum of their eigenvalues is on both sides of the imaginary

axis. This indefinite nature gets worse for increasing H and there exist to date no classical preconditioner that is able to handle that difficulty [59]. Direct solvers are of course able to invert indefinite operators. Indeed, in 2D, it is possible to show that the fill-in of a LU factorization can be reduced to $n \log n$ which means that solving very large 2D problems can be achieved using direct linear solvers. In 3D, however, the fill-in cannot be reduced under n^2 and a problem with 27 10^6 unknowns can neither be stored in the memory of state-of-the-art desktop or laptop computer nor solved in a reasonable time.

In summary, using the high-order FEM to handle geometrical complexity and heterogeneous media leads to a combination of "solving at high frequency" and "solving in the frequency domain" plus that "pollution effect", which makes a combination of harmless problems a notoriously difficult one [34]. This motivates our quest for specialized strategies that can address these issues arising from solving high-frequency Helmholtz problems.

State of the art: linear solvers for Helmholtz problems

While solving Helmholtz problems employing direct solvers is a tough challenge, researchers are still devoted to working on standard linear algebra solvers such as multi-frontal solvers [28]. Multi-frontal solvers are variants of Gauss elimination that take advantage of compressed linear algebra to avoid a large number of operations involving zero terms and gain more efficiency. While these solvers have been applied to Helmholtz problems [94, 2], they are still limited by sub-optimal complexity and memory scaling [68].

In another series of contributions, researchers focus on specialized strategies for iterative methods. The objectives of modern iterative solvers for Helmholtz problems are to derive and use solvers that lead to a number of iterations that is independent of the wavenumber [59]. To this end, one needs to remedy the deteriorated spectrum and the ill-conditioning of the linear system matrix in the high-frequency regime [60, 72, 17]. Of particular note amongst recent contributions to iterative techniques for Helmholtz problems are specialized multigrid techniques, operator-based preconditionners, and optimized domain decomposition techniques.

Multigrid methods adapted for Helmholtz problems include the wave-ray multigrid method [16], as well as Krylov subspace enhanced multigrid [29], where GMRES is proposed as a remedy for the coarse grid correction. Although multigrid methods can be used by themselves, they are more popular being used as a preconditioner for iterative solvers [22]. Preconditioning is a natural way to improve the spectral properties of the matrices arising in Helmholtz problems. Instead of traditional "matrix-based" preconditioners, which can e.g. be constructed using an incomplete LU (ILU) factorization or pseudo-inverses of the original matrix, current research is fo-

cused on "operator-based" preconditioners, which are more robust and stable. Operator-based preconditioners do not require a representation of the Helmholtz operator but are enhanced by adding an additional term with the same coefficient k^2. Complex shifted Laplacian preconditioners were proposed in [33], where an imaginary shift of the zeroth-order term was added for improved convergence rate. Because the shifted problem of the proposed preconditioners requires an expensive solver (preconditioners must be inverted exactly, e.g., by a direct solver), in practice, it is approximated by constructing ILU factors or using a multigrid method. Furthermore, to obtain good performance, one must carefully choose the shift [38, 22]. Depending on the shift and the frequency, state-of-the-art multigrid methods require a large number of iterations for the approximate inversion of the preconditioner or Krylov iterative methods require iteration numbers in general growing like $O(k^2)$ [22]. These somewhat disappointing results have led researchers to re-investigate domain decomposition (DD) methods, as an alternative to combining direct and iterative methods to solve high-frequency Helmholtz problems.

The history of DD methods dates back to 1870, one of the iterative methods for solving PDEs introduced by H.A. Schwarz [82]. There are different types of DD methods, whose general philosophy is always the same: splitting the original problem into some smaller problems on subdomains and exploit these subproblems to construct an iterative method to coordinate the solution between subdomains. According to how values at the interface are shared by subdomains, there are two families of DD methods: Schur complement methods and Schwarz methods. Schur complement methods form a reduced system with the Schur complement matrix, which is associated with interface variables. By solving the reduced system, one obtains all the interface variables. Then the remaining internal variables can be computed. FETI algorithms[3] are one family of Schur complement methods [35, 37]. The other family of domain decomposition methods, Schwarz methods, solves subproblems by taking boundary conditions based on the most recent solution obtained from adjacent subdomains. There are two families: overlapping Schwarz methods in which subdomains do overlap; and non-overlapping Schwarz methods where two neighboring subdomains communicate through an artificial interface. Compared to non-overlapping partitions, overlapping partitions possess some drawbacks. First, the size of subproblems is increased by overlapping partitions. Second, for a physical problem that can be divided into a small number of regions where the modeling equations are different (for example, the Euler equation in the far-field, the Navier Stokes equation close to the wing), overlapping strategies may cause some difficulties

[3] According to the type of domain decomposition configurations, FETI algorithms and the method of polarized traces [96] can be classified into non-overlapping DD methods [58, 88, 41].

for solving this problem. However, for non-overlapping strategies, one needs accurate boundary conditions at shared interfaces, which are non-trivial to design. In conclusion, although there are some drawbacks for overlapping strategies and non-overlapping strategies, there are many algorithms that are built successfully with both overlapping partitions and non-overlapping partitions. In this thesis, we focus on non-overlapping Schwarz methods. Non-overlapping Schwarz methods were introduced by P.-L. Lions [66] for the Laplace equation and proven to converge for the Helmholtz equation by B. Després [27]. Transmission conditions at the interface in [27] are based on a zeroth-order impedance-type operator. To improve the convergence rate of non-overlapping Schwarz methods, considerable efforts have been made to develop more efficient transmission conditions. Optimized Schwarz methods were introduced by M. J. Gander *et al.* [40], where the zeroth-order operator is replaced by a second-order approximation of the nonlocal Dirichlet-to-Neumann (DtN) map, with coefficients to be optimized a priori depending on the geometrical configuration of the subdomains and the frequency. Later, quasi-optimal optimized Schwarz methods were proposed by Y. Boubendir *et al.* [12], by using a rational approximation of the nonlocal DtN.

While Schwarz methods can handle huge, indefinite, and ill-conditioned linear systems, give a good structure to distribute computations to processors in a parallel environment, and have convergence rates that is quasi-independent of the wavenumber, Schwarz methods still do not scale with the number of subdomains. As one family of iterative methods, it is thus natural to develop robust preconditioners for Schwarz methods. For overlapping Schwarz methods, the original finite space that the problem belongs to can be decomposed into a family of subspaces, which are related to a partition into subdomains. The original space is not necessarily a direct sum of subspaces. In other words, it is possible to add a subspace, which is easy to solve, and can improve the convergence rate of the problem. This subspace is usually related to a coarse problem, often built on a coarse mesh, which is referred to as "coarse space" or "global space". The coarse space allows for capturing the global behavior of the solution, which can handle difficulties related to the heterogeneity and the number of subdomains. Generally, overlapping Schwarz methods with a coarse space are often referred to as two-level overlapping Schwarz methods. Designing coarse spaces for high-frequency Helmholtz problems is however an open problem. For non-overlapping Schwarz methods, sweeping-type preconditioners[4] are good candidates, which promise a number of iterations of DD methods that are quasi-independent of the number of subdomains. The first sweeping preconditioner which shows the quasi-independence of the number of iterations of DD meth-

[4]There are also sweeping preconditioners based on overlapping Schwarz methods [63].

ods and the number of subdomains is proposed in [32, 31]. Therefore, sweeping preconditioners garnered a lot of interest by researchers and led to some related preconditioners such as the source transfer preconditioner [18], the double sweep preconditioner [91, 92], the method of polarized traces [96], the improved sweeping domain decomposition preconditioner [85]. Disappointingly, sweeping preconditioners do not preserve the parallel properties of DD methods, as they rely on intrinsically sequential operations (they are related to an LU-type factorization of the underlying iteration operator) and they are naturally only suited for layered-type domain decompositions (where the layered structure allows to explicit the LU factorization as a double sweep across the subdomains). These drawbacks deteriorate the parallel scalability of current sweeping preconditioners. Recently, a sweeping preconditioner, called "L-sweeps", has been applied to the Helmholtz equation with a checkerboard domain partition [87]. Later, the source transfer preconditioner with a layered partition has also been extended to a diagonal sweeping preconditioner with an overlapping checkerboard partition in [63]. The parallel property of these sweeping preconditioners is attributed to the checkerboard domain decomposition, on which information can propagate in more directions. These directions of wave propagation on the checkerboard domain decomposition provide a larger space for sweeping strategies comparing to that on the layered domain decomposition.

Previous work and main contributions

In this thesis, we focus on sweeping preconditioners for non-overlapping Schwarz methods applied to Helmholtz problems. This branch of sweeping preconditioners dates back to the article of A. Vion *et al.* in 2013 [93], from which the idea to study the particular structure of the iteration matrix in the case of a layered partitioning emerged. One year later, this work was fully analyzed in [90, 91]. In 2018, the author of this thesis attended the 13th World Congress in Computational Mechanics (WCCMXIII) and listened to the talk by M. Taus who proposed L-sweeps parallel preconditioners [87] based on the method of polarized traces [96]. Our methods are inspired by the idea of M. Taus, and therefore the subject of sweeping preconditioners was revived. There is a very interesting point: the method of L-sweeps based on the method of polarized traces is inspired by sweeping preconditioners (this statement is said in [87]) and our branch of sweeping preconditioners is inspired by L-sweeps preconditioners. This shows that these two methods have a very close connection.

Our aim is to present a family of generalized sweeping preconditioners where sweeps can be done, in parallel, in several directions, for block checkerboard-type domain decompositions. This aim relies on the availabil-

ity of accurate, high-order transmission conditions between the subdomains. And the key to the accurate and efficient preconditioner that we want to propose is an accurate treatment of cross-points. Accurate, high-order transmission conditions have a long history. These transmission conditions for DD methods date back to the paper of Y. Boubendir *et al.* in 2011 [12]. In this paper, the authors propose a new "square-root" transmission operator, localized using rational approximations, which accurately approximates the DtN operator. These approximated, high-order transmission conditions allow designing an algorithm on layered domain partitions with quasi-optimal convergence properties. Some years later, this work was further developed for checkerboard domain partitions, which require careful treatment on the corner of each subdomain. This work applied to DD methods was presented in WCCMXIII in 2018 by A. Modave and the related paper was published in [71], which is based on the accurate high-order absorbing boundary conditions with corner treatments [70]. The feasibility of DD methods on checkerboard domain partitions makes generalized sweeping preconditioners possible.

The last point related to previous works is about the efficient assembly of finite element matrices, an essential building block of the proposed domain decomposition strategy for high-order FEM. The efficient implementations on assembling linear systems were proposed for discontinuous Galerkin methods based on nodal polynomials in the Ph.D. thesis of J. Lambrechts [61], and an extension to vector-valued problems and non-nodal methods using high-order FEM by N. Marsic [68]. For testing efficient finite element assembly and comparing it to the corresponding solving phase, the resulting linear system in [68] was solved using the MUMPS direct solver, and simulations are carried out on an Intel Xeon E5645 using six threads. In this thesis, we propose new implementations on modern multi-threaded architectures.

The four main contributions of this thesis are the following:

1. *Implementation of an efficient finite element method on multi-threaded architectures*
   Following the ideas initially proposed in [61, 68], we implement an efficient high-order finite element solver on modern multithreaded architectures for high-frequency Helmholtz problems. This constitutes the fundamental building block of our domain decomposition solver.

2. *Performance analysis of optimized Schwarz methods with various types of transmission conditions*
   We apply optimized Schwarz algorithms on high-order finite element discretizations on high-performance computers and analyze their performance on layered domain decomposition configurations with different transmission conditions.

3. *A family of generalized sweeping preconditioners*
   We generalize sweeping preconditioners from layered-type decompositions to block (checkerboard- and Rubik's cube-type) decompositions, which can be efficiently parallelized.

4. *Improved parallelization strategy of sweeping preconditioners for multiple right-hand sides*
   We propose to implement parallel sweeping preconditioners for multiple right-hand sides, based on pipelining algorithms and a variant of parallel sweeping preconditioners. This implementation for multiple right-hand sides is of great practical value considering that the study of physical problems involving the propagation of waves, like seismology, usually involves a lot of right-hand sides linked to the presence of multiple wave sources.

A finite element program has been developed from scratch using the C language.[5] Several resources have been used to develop our original code:

1. Gmsh [43] is used to handle mesh generation, domain decomposition, and post-processing;

2. OpenBLAS [95], an optimized BLAS (Basic Linear Algebra Subprograms) library based on the GotoBLAS2 1.13BSD version, is used to deal with elementary dense linear algebra;

3. The direct solver Pardiso [26, 89, 57] and the MKL library [53] are used to solve the sparse linear system generated on each subdomain.

Outline

- In Chapter 1, we review the finite element method and variational formulations for acoustic wave problems. Next, we detail the efficient assembly procedure, based on an efficient quadrature approach on multithreaded computers.

- In Chapter 2, the non-overlapping DD methods are presented. We analyze the behavior of these methods with layered DD configurations, including memory usage, peak performance, timing, and iteration count. For layered configurations, we propose some performance improvements, analyze the convergence rate with various types of transmission conditions, and discuss the limitations. Next, the non-overlapping DD methods with checkerboard configurations in two dimensions and Rubik's cube configurations in three dimensions are presented. For these

[5] For original codes, please contact UCLouvain Hextreme team.

configurations, there are special treatments required for cross-points where more than two subdomains meet [71]. These special treatments determine the convergence rate of the DD methods, which we study for scattering problems in free space and for waveguide-type problems.

- In Chapter 3, we propose generalized sweeping preconditioners for non-overlapping DD methods with checkerboard and Rubik's cube DD configurations. We present various numerical experiments to test the performance of our proposed preconditioners.
- In Chapter 4, considering that the research of physical problems involving the propagation of waves which have multiple sources, we propose some strategies for our proposed sweeping preconditioners for multiple right-hand sides.
- Finally, we draw some conclusions and perspectives for future research.

Efficient finite element method on multi-threaded architectures

1

In this chapter, we describe an efficient implementation of the high-order finite element method for Helmholtz problems, inspired by the seminal work from [61, 68]. We explore the efficiency of the resulting algorithm and discuss its limitations on modern multi-threaded architectures, both in terms of the matrix assembly and the (direct) solution of the linear system.

1.1 Variational form of Helmholtz problems

We consider a wave propagation problem in the frequency domain posed in an heterogeneous medium $\Omega \in \mathbb{R}^d$ for $d = 2$ or 3, i.e. we aim at finding the solution $u \in H^1(\Omega)$ solution to

$$\begin{cases} \Delta u + k^2(\boldsymbol{x})\, u = 0 & \text{in } \Omega, \\ u = f & \text{in } \Gamma_{\text{src}}, \end{cases} \tag{1.1}$$

with Sommerfeld radiation condition

$$\lim_{r\to\infty} \int_{\partial B_r} \left| \frac{\partial u}{\partial r} - i\, k_0\, u \right|^2 d\partial B_r = 0. \tag{1.2}$$

where $k(\boldsymbol{x}) = k_0$ for $|\boldsymbol{x}| > r_0$ ($r_0 > 0$), Γ_{src} is a boundary where a Dirichlet boundary condition is prescribed, and B_r refers to the ball with radius r centered at the origin. In order to solve this problem using the FEM, one truncates the domain at some finite distance, and at this artificial boundary Γ_∞ one imposes an absorbing boundary condition. In this chapter, to find a solution $u \in H^1(\Omega_+)$ in the truncated domain Ω_+, we solve the following problem using zeroth-order impedance boundary conditions on Γ_∞:

$$\begin{cases} \Delta u + k^2\, u = 0 & \text{in } \Omega_+, \\ u = f & \text{on } \Gamma_{\text{src}}, \\ \dfrac{\partial u}{\partial r} - iku = 0 & \text{on } \Gamma_\infty. \end{cases} \tag{1.3}$$

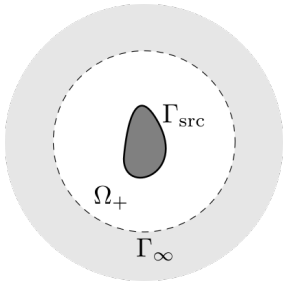


FIGURE 1.1: Free space scattering model, and definition of boundaries. Gray regions refer to the domain Ω that extends to the infinity, and the white region refers to the computational domain Ω_+ truncated by the artificial boundary Γ_∞ (dashed line).

The standard weak form of (1.3) writes [83]: find $u \in H^1(\Omega_+)$ with $u = f$ on Γ_{src} such that

$$a(u, v) = 0 \quad \forall v \in H_0^1(\Omega_+), \tag{1.4}$$

where

$$a(u, v) := \int_{\Omega_+} \left(\nabla u \cdot \overline{\nabla v} - k^2 u \bar{v} \right) d\Omega_+ - \int_{\Gamma_\infty} ik\, u \bar{v}\, d\Gamma_\infty \tag{1.5}$$

and where $\bar{x}$ denotes the complex conjugate of x.

1.2 Finite element discretization

Let us focus on the volume part of the variational form (1.5):

$$\int_{\Omega_+} \nabla u \cdot \overline{\nabla v}\, d\Omega_+ - \int_{\Omega_+} k^2\, u\, \overline{v}\, d\Omega_+ , \qquad \forall v \in H_0^1(\Omega_+) .$$

After discretization of Ω_+ with a finite element mesh $\mathcal{T}_h$ with N nodes, applying a Galerkin discretization to this volume part with Lagrange elements (H^1-conforming finite elements) [83, 68] leads to a sum of elementary integrals over every element $K \in \mathcal{T}_h$:

$$\sum_{j=0}^{N-1} \sum_{K \in \mathcal{T}_h} \left(\int_K \nabla u_j \cdot \overline{\nabla v_i}\, dK - \int_K k^2\, u_j\, \overline{v_i}\, dK \right) , \qquad \forall v_i \in H_0^1(K) ,$$

where i and j refer to the indices of the basis functions $v_i \in H_0^1(\Omega_+)$ and $u_j \in H^1(\Omega_+)$, respectively. For each element K, the basis functions are classically expressed in terms of basis functions in a reference element obtained through

a mapping $\Phi_K : \hat{K} \longrightarrow K$. Thus, each elemental integral can be computed as:

$$\sum_{j=0}^{N-1} \sum_{K \in \mathcal{T}_h} \left(\int_{\hat{K}} \left(J_K^{-T} \nabla \phi_{l(j)} \right) \cdot \left(J_K^{-T} \overline{\nabla \phi_{k(i)}} \right) |\det J_K| \; d\hat{K} - \int_{\hat{K}} k^2 \, \phi_{l(j)} \, \overline{\phi_{k(i)}} \; |\det J_K| \; d\hat{K} \right), \qquad \forall \phi_{k(i)} \in H_0^1(\hat{K}) \,, \tag{1.6}$$

where $\phi_{l(j)}$ and $\phi_{k(i)}$ denote the shape functions on the reference element. Here, we have new indices $l(j)$ and $k(i)$, where l and k are local indices in the reference element and j and i are global indices in the "physical" element in the mesh. The notations $k(i)$ or $l(j)$ mean that the local indices k and l in the reference element are mapped onto global indices i or j in the physical element.

1.3 Efficient quadrature based on multi-threaded architectures

Quadrature methods are approximations of the numerical value of the definite integral of a function. Generally, one replaces the computation of the integral by a weighted sum of the value of the function to integrate at some specified points. In one dimension, the Gaussian quadrature method, named after Carl Friedrich Gauss, is an exact quadrature for a polynomial of degree $2n - 1$ with n points of integration. Exact quadrature rules can be designed in higher dimensions as well, which play a fundamental role in the finite element method.

Consider a polynomial function defined on a reference element $\hat{K} \in \mathbb{R}^d$, the integral of this polynomial

$$I = \int_{\hat{K}} p(\boldsymbol{x}) d\hat{K}$$

can be computed exactly as a weighted sum of this function, called a quadrature:

$$I = \sum_{\hat{K}}^{G} w_g p(\boldsymbol{x}_g) d\hat{K} \,,$$

where, $w_g \in \mathbb{R}$ and $\boldsymbol{x}_g \in \hat{K}$ are well chosen weights and evaluation points, respectively.

1.3.1 Efficient quadrature

In a traditional finite element algorithm, the elementary integrals are computed on the fly by fetching corresponding shape functions and Jacobians

to assemble the discretized system. An efficient quadrature procedure using matrix multiplication has been developed in [68] for arbitrary higher-order polynomial finite element method, as an extension of the method proposed in [50, 51, 61] for discontinuous Galerkin schemes with high-order Lagrange elements. The underlying idea behind this efficient assembly procedure is to compute all the elementary terms "at once" using matrix multiplication, which can be highly optimized on multi-core computer architectures and are standardized through the Basic Linear Algebra Subprograms (BLAS) programming interface [95]. Optimized BLAS implementations provide efficient use of CPU registers and efficient cache reuse. Moreover, modern processors like the Intel Knights Landing (KNL) architecture [53] that we have extensively used in our tests, support Intel AVX-512 instructions, which allow further optimizations through vectorization for matrix multiplications.

To achieve efficient matrix operations, terms of the integrals of (1.6) have to be re-organized. In what follows we detail how the two main terms required for Helmholtz problems, involving respectively products of H^1 and $H(\mathrm{curl})$ functions, are treated, following the strategy presented in [68].

1.3.1.1 Products of H^1 functions

Let us first consider the second term in the weak form of the time-harmonic acoustic wave equation (1.6). This term involves the integral over an element K in reference coordinates and the local degrees of freedom k and l, as well as a (possibly space-dependent) function $\alpha = k^2$:

$$\mathcal{I}_{K,\{k,l\}} = \int_{\hat{K}} \left(\alpha\, \phi_l\right)\left(\overline{\phi_k}\right) |\mathrm{det}\boldsymbol{J}_K|\, d\hat{K} \,. \tag{1.7}$$

This integral can be re-organized as follows:

$$\begin{aligned}
\mathcal{I}_{K,\{k,l\}} &= \sum_{g=1}^{G} w_g \alpha\Big|_g \phi_l\Big|_g \overline{\phi_k}\Big|_g \,|\mathrm{det}\boldsymbol{J}_K|\,\Big|_g \\
&= \sum_{g=1}^{G} \left(\alpha\Big|_g \,|\mathrm{det}\boldsymbol{J}_K|\,\Big|_g\right)\left(w_g \phi_l\Big|_g \overline{\phi_k}\Big|_g\right) \\
&= \sum_{g=1}^{G} \boldsymbol{B}\Big[K, g\Big]\boldsymbol{C}\Big[g, \{k, l\}\Big] \\
&= \boldsymbol{D}\Big[K, \{k, l\}\Big] \,.
\end{aligned} \tag{1.8}$$

From this last equation, the integral $\mathcal{I}_{K,\{k,l\}}$ over an element K and the local degrees of freedom k and l can be obtained by the product of two matrices $\boldsymbol{B}$ and $\boldsymbol{C}$. Matrix $\boldsymbol{B}$ consists of the evaluation at quadrature points of the

function α and the determinant of the Jacobian of element K, which is only dependent on the mesh data. And matrix $\boldsymbol{C}$ consists of weights and the evaluation at quadrature points of shape functions, which is only dependent on the reference space $\hat{K}$.

1.3.1.2 Products of $H(\mathrm{curl})$ functions

The first term in (1.6) involves the following integral $\mathcal{I}_{K,\{k,l\}}$ over an element K in the reference coordinates and the local degrees of freedom k and l:

$$\mathcal{I}_{K,\{k,l\}} = \int_{\hat{K}} \left(\boldsymbol{J}_K^{-T}\nabla\phi_l\right)\left(\boldsymbol{J}_K^{-T}\overline{\nabla\phi_k}\right)|\mathrm{det}\boldsymbol{J}_K|\, d\hat{K}\ . \tag{1.9}$$

This integral can be re-organized as follows:

$$\begin{aligned}
\mathcal{I}_{K,\{k,l\}} &= \sum_{g=1}^{G} w_g \sum_{a=1}^{3}\left(\sum_{b=1}^{3}(\boldsymbol{J}_K)^{-T}_{a,b}\Big|_g(\nabla\phi_l)_b\Big|_g\right)\left(\sum_{c=1}^{3}(\boldsymbol{J}_K)^{-T}_{a,c}\Big|_g(\overline{\nabla\phi_k})_c\Big|_g\right)|\mathrm{det}\boldsymbol{J}_K|\,\Big|_g \\
&= \sum_{g=1}^{G} w_g \sum_{a=1}^{3}\sum_{b=1}^{3}\sum_{c=1}^{3}(\boldsymbol{J}_K)^{-T}_{a,b}\Big|_g(\boldsymbol{J}_K)^{-T}_{a,c}\Big|_g(\nabla\phi_l)_b\Big|_g(\overline{\nabla\phi_k})_c\Big|_g|\mathrm{det}\boldsymbol{J}_K|\,\Big|_g \\
&= \sum_{g=1}^{G}\sum_{b=1}^{3}\sum_{c=1}^{3}\left(|\mathrm{det}\boldsymbol{J}_K|\,\Big|_g\sum_{a=1}^{3}(\boldsymbol{J}_K)^{-T}_{a,b}\Big|_g(\boldsymbol{J}_K)^{-T}_{a,c}\Big|_g\right)\left(w_g(\nabla\phi_l)_b\Big|_g(\overline{\nabla\phi_k})_c\Big|_g\right) \\
&= \sum_{g=1}^{G}\sum_{b=1}^{3}\sum_{c=1}^{3}\boldsymbol{B}\Big[K,\{g,b,c\}\Big]\boldsymbol{C}\Big[\{g,b,c\},\{k,l\}\Big] \\
&= \boldsymbol{D}\Big[K,\{k,l\}\Big]\ .
\end{aligned} \tag{1.10}$$

The integral $\mathcal{I}_{K,\{k,l\}}$ can thus be obtained by the product of two matrices $\boldsymbol{B}$ and $\boldsymbol{C}$. Matrix $\boldsymbol{B}$ consists of the evaluation at quadrature points of the Jacobian of the element K, which is only dependent on the mesh data. And matrix $\boldsymbol{C}$ consists of weights and the evaluation at quadrature points of shape functions, which is only dependent on the reference space $\hat{K}$.

1.3.2 Matrix operations on general multi-core processors

As mentioned above, we aim at exploiting the efficiency of matrix multiplication offered by optimized BLAS implementations on modern multi-core architectures (e.g. Intel Xeon Phi or AMD EPYC) to speed-up the assembly of our finite elements matrices. In this section, we review the principle of matrix-matrix multiplication implemented on multi-core processors, which is proposed in [48, 42] and detail implementation choices on the Intel Xeon Phi [49, 64]. We focus on double complex precision matrix-matrix multiplication, i.e.

Algorithm 1: Matrix-matrix multiplication algorithm on multi-core processor.

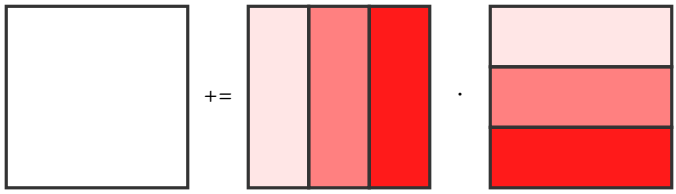
```
for p = 0 : k − 1 in steps of k_b do
    Pack B(p : p + k_b − 1, 0 : n − 1) into B̃ ;
    for q = 0 : m − 1 in steps of m_b do
        Pack A(q : q + m_b − 1, p : p + k_b − 1) into Ã ;
        for jr = 0 : n − 1 in steps of n_r do
            B̂ = B̃(:, jr : jr + n_r − 1) ;
            for ir = 0 : m_b − 1 in steps of m_r do
                Â = Ã(ir : ir + m_r − 1, :) ;
                Ĉ+ = Â × B̂ ;
                Update C using Ĉ ;
            end
        end
    end
end
```

the so-called ZGEMM BLAS routine: $C = \alpha AB + \beta C$, where A, B and C are $m \times k$, $k \times n$ and $m \times n$ matrices, respectively. In what follows it is assumed that the matrices are stored in row-major order. In [48], Gunnels et al. propose a family of algorithms for matrix-matrix multiplication on hierarchical memory architectures. These algorithms attempt to amortize the cost of moving data between memory layers. In [42], Goto et al. have given a systematic analysis of the high-level issues that affect the design of high-performance matrix-matrix multiplications which were proposed in [48] and argued that Algorithm 1 is ideal for the implementation of matrix-matrix multiplication.

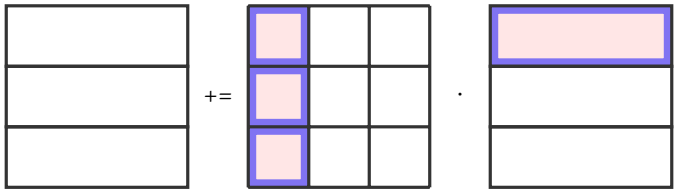
Figure 1.2a, 1.2b, 1.2c, and 1.2d illustrate Algorithm 1. The main idea of this algorithm is to break matrix-matrix multiplication into a sequence of outer products. Furthermore, this algorithm blocks the matrices in the sequence of outer products to fit into one or more levels of caches on a given architecture.

In the following parts, we will describe this implementation in three parts: reuse of cache, use of full vector registers, and bandwidth of the memory.

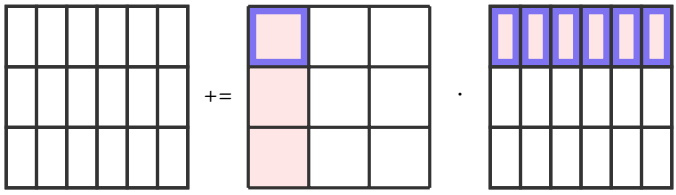
Reuse of cache: Highly optimized implementations of general matrix multiplication block the matrices to fit into multi-levels of cache on a given architecture. If the block matrices fit into the cache, they can be reused during computation with fast accesses. Therefore, the bandwidth requirements are reduced to be under the limit of the architecture can deliver. Next, we discuss the size of block matrices. In [42], Goto and van de Geijn proposed



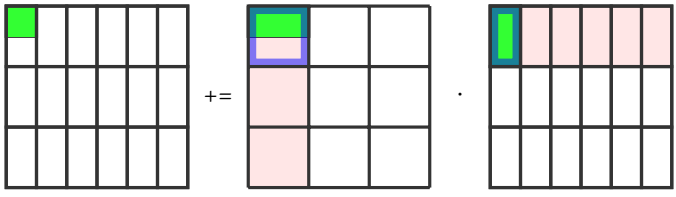
(A) Illustration of p-loop in Algorithm 1. Matrix-matrix multiplication is broken into a sequence of outer products. In this illustration, broken row and broken column of same color have their outer product.



(B) Illustration of q-loop in Algorithm 1. Each broken row of matrix A in Figure 1.2a is divided into smaller blocks, which have outer products with corresponding broken column in matrix B.



(C) Illustration of jr-loop in Algorithm 1. Each broken column of matrix B in Figure 1.2a is divided into smaller blocks, which have outer products with corresponding broken row in matrix A (Here, broken row in matrix A has already divided into smaller blocks).



(D) Illustration of ir-loop in Algorithm 1. Each small block in broken row of matrix A is divided into smaller matrices (green part in matrix A) in order to fit in register.

FIGURE 1.2: Illustration of Algorithm 1.

the following ideas:

- Typically half the available registers are used for the storing $m_r \times n_r$ matrix $\hat{C}$. Thus, we have

$$m_r \times n_r \times 16\,\text{bytes} \approx \frac{\text{size of L1}}{2};$$

- To amortize the cost of updating $m_r \times n_r$ elements of $\hat{C}$, k_b should be picked to be as large as possible;

- Matrix $\hat{B}$ are reused many times, and therefore must remain in the L1 cache. In practice, $\hat{B}$ should occupy less than half of the L1 cache so that matrices $\hat{A}$ and $\hat{C}$ do not evict matrix $\hat{B}$ This gives

$$k_b \times n_r \times 16\,\text{bytes} \leq \frac{\text{size of L1}}{2};$$

- In practice, m_b is typically chosen so that $\tilde{A}$ only occupies about half of the memory addressable by the TLB and the L2 cache. This gives

$$m_b \times k_b \times 16\,\text{bytes} \leq \frac{\text{size of L2}}{2}.$$

Roughly, according to the ideas of Goto and van de Geijn, we require matrices $\tilde{A}$, $\hat{B}$ and $\hat{C}$ to fit into the L2 cache. This is the reuse of cache for general multi-core architecture.

In this thesis, optimized BLAS implementations based on the aforementioned principles were used on the Intel Xeon Phi 7210 and AMD EPYC 7551 processors. Below we present some specific strategies used on the Intel Xeon Phi thanks to the Intel AVX-512 instructions, and some combinations of multi-core computers and linear algebra libraries.

1.3.3 Matrix operations on the Intel Xeon Phi processor with AVX-512

Lut us consider the specifics of the Intel Knights Landing (KNL) architecture. The KNL design has a concept of a tile, which is the basic unit for replication. Each tile consists of two cores, two vector-processing units (VPUs) per core, and a 1 MB L2 cache shared between the two cores. KNL can reach peak performance with one, two, or four threads [53]. In this work, we use one thread per core which is enough to reach maximum performance. In [64], the authors indicate that the reuse of cache L1 of matrix $\hat{B}$ is not essential and L1 cache blocking is not crucial for implementing general matrix multiplication on the KNL. They also show that a good performance is obtained when matrix

$\tilde{A}$, $\hat{B}$ and $\hat{C}$ occupy about a quarter of the L2 shared cache by two cores. Thus, we have following inequality:

$$(m_b k_b + k_b n_r + m_r n_r) \times 16 \text{ bytes} \leq 256 \text{ KB} .$$

Use of full vector registers: KNL has emerged with the Intel AVX-512 vector instructions. High performance can be achieved with the use of full vector registers [53]. Given that AVX-512 instructions provide 512-bit vector operations, $m_r \times n_r$ matrix $\hat{C}$ must be well organized on the L1 cache. n_r should be a multiple of 4 and the inner kernel uses $m_r \times n_r/4$ vector registers to store the value of $\hat{A} \times \hat{B}$. There are 32 vector registers per core on the KNL. Thus, we have:

$$\frac{n_r}{4}(1 + m_r) \leq 32 ,$$

where $n_r \bmod 4 = 0$. The following pairs of register block sizes (m_r, n_r) can be considered:

$$(m_r, n_r) = (31, 4), (15, 8), \text{ or } (7, 16) .$$

Bandwidth of memory: When implementing general matrix multiplication on the KNL, we should consider the bandwidth of the memory. The required memory bandwidth for computing $m_b \times n$ block is the ratio between the memory traffic and computation time. To update $m_b \times n$ size of the block of matrix C, there are $(m_b n + m_b k_b + k_b n) \times 16$ bytes of data moving from the main memory to the L2 cache memory. Meanwhile, there are $(m_b n) \times 16$ bytes of data moving from the registers to the main memory for ZGEMM. Considering that each tile in KNL consists of two VPUs and the inner kernel uses the fused-multiply add instruction, it requires $(m_b \times n \times k_b)/16$ cycles to calculate size $m_b \times n$ block of matrix C. We can obtain the required memory bandwidth:

$$256\,(2/k_b + 1/n + 1/m_b) .$$

One thing that we must pay attention to is that memory latency increases with the bandwidth of the memory. We must resize the cache block sizes m_b and k_b to reduce memory latency. In [49], m_b is set to be about 120 for the design of general matrix multiplication based on Intel Xeon Phi processor, and in [64], m_b is set to be 124 for the implementation of matrix multiplication on the Intel KNL processor with AVX-512. Therefore, we will consider $m_b \approx 120$ for the following discussions.

We have introduced the basic idea of the implementation of ZGEMM on the Intel Xeon Phi processor. According to this idea, we can determine the cache block sizes which are designed for Intel Xeon Phi in [49]. Finally, we must explain that we infer from [49, 64] that the cache block size for ZGEMM on KNL is probably designed similarily to Table 1.1.

(m_r, n_r)	(31, 4)	(15, 8)	(7, 16)
No. of vector registers	32	32	32
k_b	120	110	110
m_b	124	135	126
Size of $(\tilde{A} + \hat{B} + \hat{C})$	242 KB	248 KB	246 KB

Table 1.1: Cache block size for ZGEMM on KNL.

1.3.4 Hardware and software descriptions

All the software developments and the numerical tests for this thesis were performed on both the Intel Xeon Phi 7210 and the AMD EPYC 7551 processors (see Table 1.2).

The Intel Xeon Phi is designed to reach the ability to fully utilize the scaling capabilities of Intel Xeon processor-based systems [53]. KNL offers the ability to make a system that can potentially offer exceptional performance while still being buildable and power-efficient [53]. For instance, the matrix multiplication function DGEMM optimized by Intel Math Kernel Library for Intel Xeon Phi Processors can reach 4500 GFLOPS/s with 68 cores.

AMD EPYC 7551 is designed with four system-on-chip (SoC) dies which are interconnected using the Infinity Fabric. Each SoC has 8 cores within a single CPU package. Although the cores are separated into several packages, the Infinity Fabric designed by AMD can minimize core-to-core communication and connect up to 2 sockets, that is a 64-core supercomputer, which enables immense memory capacity. For instance, an AMD EPYC 7551 with two sockets supports up to 512 GB of RAM, i.e. much more than that of Intel Xeon Phi. Furthermore, this architecture can support much more bandwidth with a single socket. It also provides linear scalability as the socket increases.

64-cores computer	Intel Xeon Phi 7210	AMD EPYC 7551
Base frequency	1.3 GHz	2.0 GHz
Cache size	1024 KB	512 KB
RAM	201 GB	515 GB

Table 1.2: Configuration details of processors.

The following formula estimates the theoretical peak GFLOPS of processors:

$$\text{Peak GFLOPS} = \text{Num. of Cores} \times \text{Avg. GHz} \times \text{Operations per cycle}. \quad (1.11)$$

The average GHz of Xeon Phi 7210 is 1.3 GHz and that of AMD EPYC is 2.0

GHz. Thus, the theoretical GFLOPS for Intel Xeon Phi and AMD EPYC are listed below:

Architecture	Precision	OPC	Peak GFLOPS
64 cores Xeon Phi 7210	ZP	16	1331.2
64 cores AMD EPYC 7551	ZP	8	1024.0

Table 1.3: Theoretical GFLOPS for 64-cores processors.

There are some popular libraries that implement sparse direct solvers, such as PARDISO [26, 89, 57]. Intel implements an old version of PARDISO in its own library MKL [53], which optimizes features and maximizes performance. The latest version PARDISO improves the parallelism on forward and backward substitutions, which makes iterative methods much powerful. In this thesis, OpenBLAS and MKL BLAS are chosen as linear algebra libraries and MKL sparse direct solver and PARDISO 6.0.0 are chosen as direct solvers.

	Test 1	Test 2	Test 3	Test 4
Processor	Xeon Phi	Xeon Phi	EPYC 7551	EPYC
Linear Algebra	MKL BLAS	OpenBLAS	MKL BLAS	OpenBLAS
Direct solver	MKL	PARDISO	MKL	PARDISO

Table 1.4: Four combinations of hardware and software using OpenMP.

1.4 Efficient assembly

Using the quadrature methodology based on matrix multiplications allows to replace the classical on-the-fly finite element matrix assembly procedure summarized in Algorithm 2 with the efficient assembly procedure outlined in 3.

In this efficient assembly algorithm, one precomputes all the integrals by matrix-matrix multiplication, then one fetches a matrix block associated to a finite element and puts it into the global matrix. There are two differences in Algorithm 3 compared to the original efficient assembly proposed [68]: the products of $H(\text{curl})$ and H^1 functions are added using matrix operations instead of one-by-one addition (for example, one calculates the global matrix D: $D = \alpha AB + \beta D$, where βD could be the product $H(\text{curl})$, and αAB coulde be the product H^1 added to the matrix βD); and the assembly of the precomputed matrix into the global matrix is done element by element instead of data by

Algorithm 2: Classical on-the-fly strategy for the finite element assembly.

```
Function getFETerm(element e, int k, int l):
    get shape functions for Element e;
    compute elementary integral (1.7) or (1.9) associated to the
     triplet {e, k, l};
    return the integral;
EndFunction
Function assemble(vector of elements, matrix A):
    ⋮
    for e = 0; e < the number of elements; e++ do
        for k = 0; k < the number of unknowns of element e; k++
         do
            for l = 0; l < the number of unknowns of element e; l++
             do
                getFETerm(element e, k, l) added to the matrix A;
            end
        end
    end
EndFunction
```

Algorithm 3: Proposed efficient quadrature strategy for the finite element assembly.

```
Function precompute(vector of elements, matrix D):
    compute the product in (1.8) and (1.10) and add them in the
     matrix D;
    return the matrix D;
EndFunction
Function assemble(vector of elements, matrix A, matrix D):
    ⋮
    for e = 0; e < the number of elements; e++ do
        get the matrix block in D associated to the element e;
        add the matrix block to the matrix A;
    end
EndFunction
```

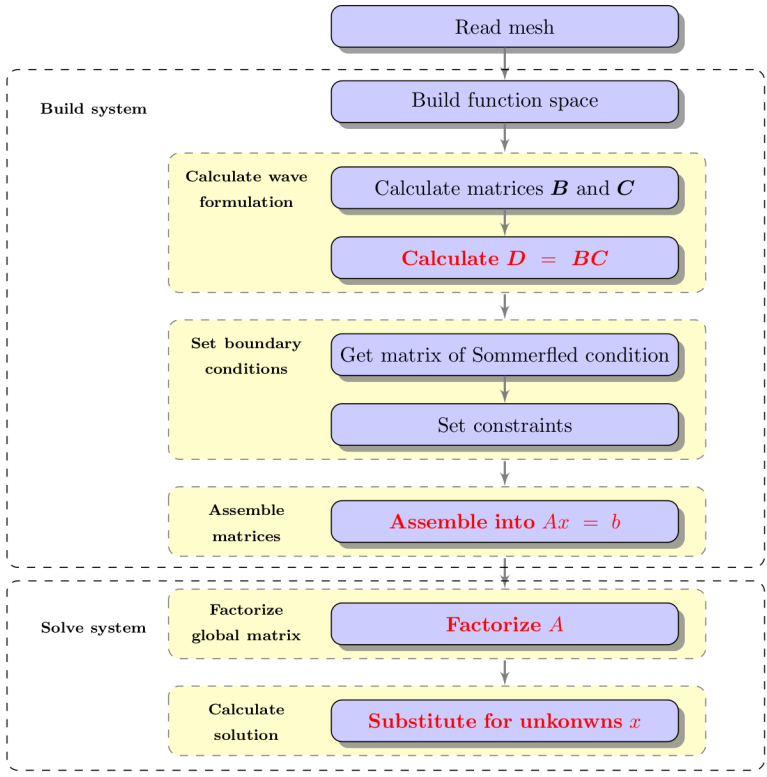


FIGURE 1.3: Numerical procedure for Helmholtz problems using direct solvers.

data. In order to achieve this, one precomputes the globalIDs for the data in each element.

1.5 Performance

In this section, the performance of the efficient assembly procedure is analyzed for representative time-harmonic acoustic wave simulations.

1.5.1 Numerical procedure

Figure 1.3 shows the general numerical procedure of the time-harmonic wave solver, which contains two main phases: assembling the linear system ("Build system") and solving the system ("Solve system"). Within these two main phases, the parts that are parallelized are written in red, and constitute the points that are under discussion. In the "Build system" phase, the parts that consume the most time are matrix-matrix products and assembling the whole

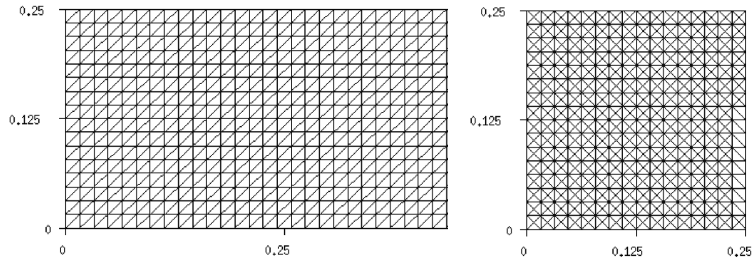


FIGURE 1.4: Example of mesh used for the performance tests on multi-core processors.

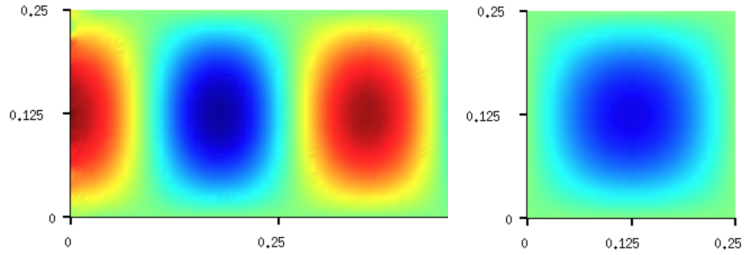


FIGURE 1.5: Example of numerical solution for the performance tests on multi-core processors.

matrix of the system. In sequential mode, building the system takes at least 95% of time. In the "Solve system" phase, the main time consuming part is the numerical factorization, which is characterized by a high memory consumption and a large number of computations, and where the modern linear algebra libraries used achieve promising parallel efficiency. The performance of forward and backward substitution will be also measured, as they will play a key role in the next chapters when the direct solvers will be used within iterative domain decomposition methods.

1.5.2 Test case and mesh

The case used to assess the performance of high-order finite element solver combining efficient assembly procedure and sparse direct solvers is a three-dimensional acoustic waveguide with wavenumber 25 rad/m. The model length is 0.5 m and cross-section of 0.25×0.25 m^2. The geometry is meshed with 16 tetrahedra per wavelength. We consider polynomial orders ranging from 1 to 7.

Type	Num of elems	order	Size of matrix	non-zeros	non-zeros(%)
Tet	47616	1	6975	9.6×10^4	0.1966
		2	59582	1.6×10^6	0.0446
		3	205437	9.3×10^6	0.0221
		4	492156	3.5×10^7	0.0145
		5	967355	1.0×10^8	0.0108
		6	1678650	2.5×10^8	0.0088
		7	2673657	5.3×10^8	0.0075

Table 1.5: Mesh and matrix of system testing on 64-core processors.

order	1	2	3	4	5	6	7
M	47616	47616	47616	47616	47616	47616	47616
N	16	100	400	1225	3136	7056	14400
K	36	36	99	216	387	1134	1890

Table 1.6: Size of stiffness matrix-matrix operations on multi-core processors.

1.5.3 Benchmarks

Let us first analyse the performance of complex double precision matrix-matrix multiplications on the Intel Xeon Phi and the AMD EPYC, in the "Build system" phase of the time-harmonic wave solver: see figures 1.6 and 1.7. While the performance of the matrix-matrix operations is quite disappointing for low-order polynomial orders (smaller than 3), the matrix-matrix operations reach the theoretical peak GFLOPS with high-order polynomials driven both by the Intel Math Kernel Library on the Intel Xeon Phi and by OpenBLAS on the AMD EPYC. Performance is significantly degraded (200 GFLOPS instead of peak 1331 GFLOPS) when using OpenBLAS on Xeon Phi, as is the case when using MKL BLAS (500 GFLOPS instead of peak 1024 GFLOPS) on AMD EPYC.

We now turn to the other parallelized part in the "Build system" phase, i.e. assembling the global matrix. Given that the previous tests reach quasi-peak GFLOPS with more than 4^{th} order polynomials, we use 7^{th} order polynomials for all subsequent tests. The assembly code takes advantage of parallelizing for loops with OpenMP and prevents the possibility of multiple, simultaneous reading and writing of threads with OpenMP atomic constructs. Although the parallel efficiency of the global matrix assembly is slightly degraded due to synchronization, Figure 1.8 shows decent performance both on Xeon Phi and

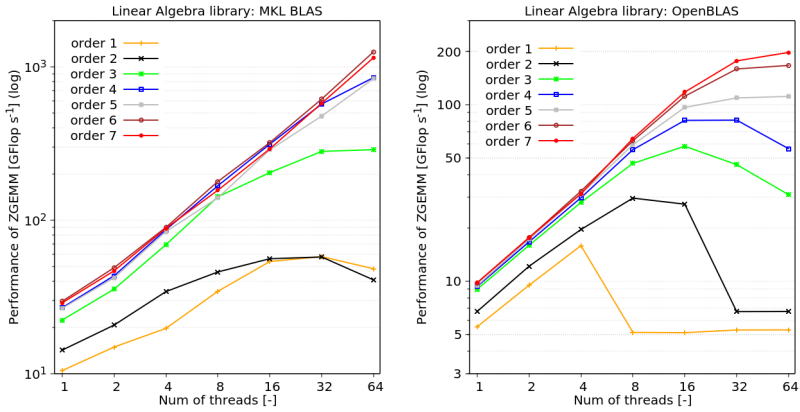


FIGURE 1.6: Matrix-matrix product (ZGEMM) on Intel Xeon Phi.

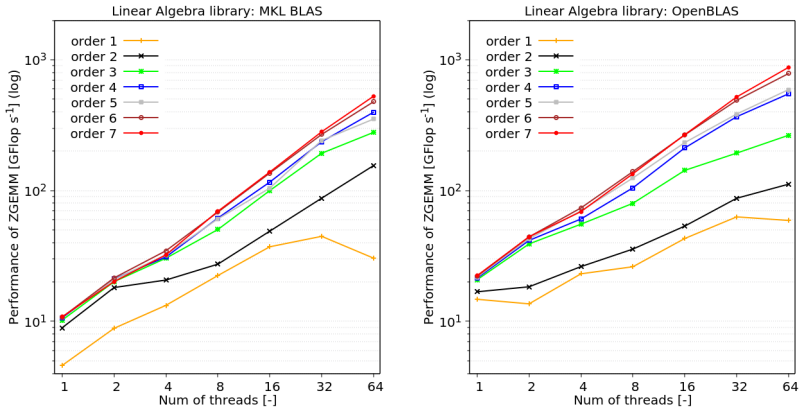


FIGURE 1.7: Matrix-matrix product (ZGEMM) on AMD EPYC.

EPYC.

After assembling the global matrix, it is time to solve the resulting linear system. From Figure 1.9, it is clear that the fastest sparse direct solver on the Xeon Phi is PARDISO 6.0.0, followed by the MKL. Both show excellent parallelism in the factorization phase. The timing of the forward and backward substitution is also reported, as it plays an important role in domain decomposition methods, where the same factorization is reused with many right hand sides during the iterative phase of the method. The parallel efficiency of PARDISO 6.0.0 is adequate during the substitution phase with 4 threads on both multi-core machines, whereas parallelism in not exploited during the substitution phase by the MKL. No matter what library is used, the performance of

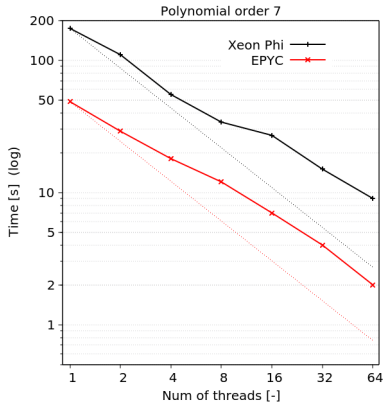


FIGURE 1.8: Global system assembly with OpenMP.

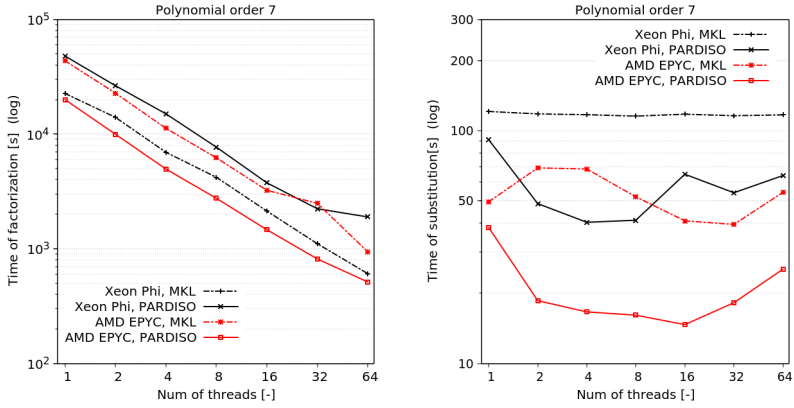


FIGURE 1.9: Linear system solution using a direct sparse solver.

substitution on EPYC is better than that on Xeon Phi. EPYC with PARDISO 6.0.0 shows the best performance and parallelism for all combinations for the substitution part.

The total solution time with direct solvers on Xeon Phi and EPYC is reported on Figure 1.10. From a pure computing time point of view, overall the AMD EPYC with PARDISO 6.0.0 has the best performance, while the Xeon Phi with the Intel MKL shows adequate performance.

To assess the overall parallel performance of solver, let us define the parallel efficiency $E(N_p)$ obtained with N_p processors as:

$$E(N_p) = \frac{1}{N_p} \frac{T(1)}{T(N_p)} , \tag{1.12}$$

$T(1)$ is the time taken by the algorithm on a single processor and $T(N_p)$ is time

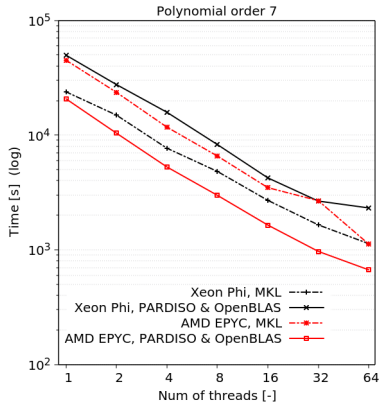


FIGURE 1.10: Total solution times.

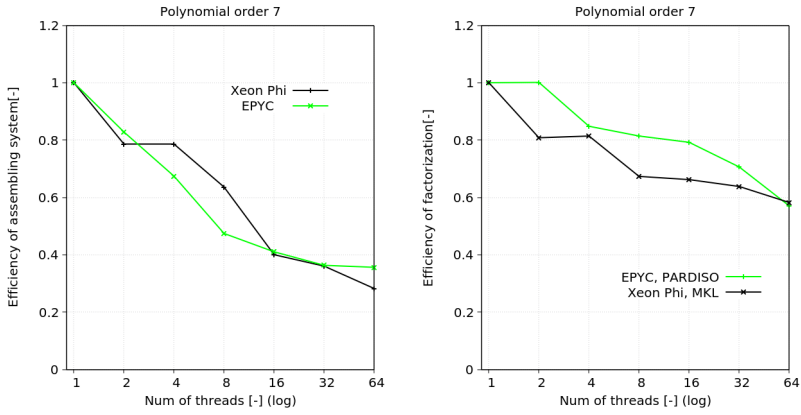


FIGURE 1.11: Parallel efficiency of direct solvers.

taken on N_p processors. Figure 1.11 reports the parallel efficiencies for the assembly and the factorization of the global matrix, which are both adequate.

The ratio between the computational time required by the "Solve system" and "Build system" phases is reported next in Figure 1.12. Compared to the naive "on-the-fly" assembly, we clearly see that for increasing polynomial orders the solution phase now clearly dominates the assembly time, as was sought. Finally, the memory required by the factorization of the global matrix is reported in Figure 1.13, which highlights the memory cost of increasing the polynomial order.

In the next chapters, we will see that the use of domain decomposition methods provides the solution to these two problems.

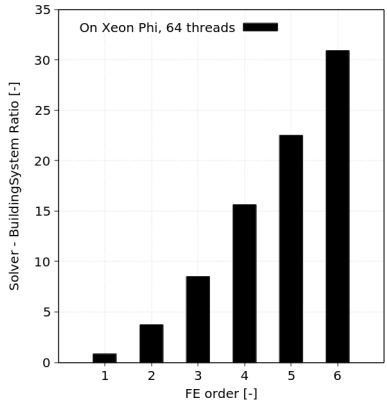


FIGURE 1.12: Ratio between the "Solve system" and "Build system" phases.

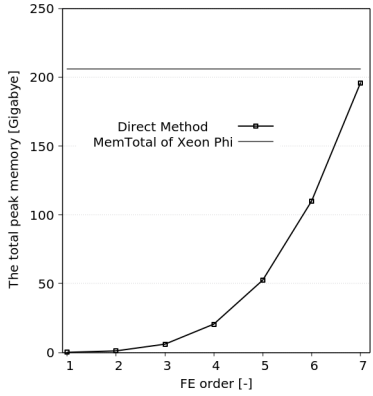


FIGURE 1.13: Memory usage of direct solvers.

1.6 Conclusion

In this chapter we have presented and analyzed an efficient finite element assembly procedure, based on the ideas proposed in [61, 68]. Together with the use of optimized sparse direct solvers for the two computing architectures of interest (Intel Xeon Phi and the AMD EPYC) the resulting Helmholtz solver constitutes the basic building block of the optimized domain decomposition methods that will be developed in the next chapters. Compared to [61, 68], we improved the efficiency of the product of $H(\text{curl})$ and H^1 functions are added, and how the precomputed matrices are assembled in the global matrix. Matrix-matrix multiplications for the high-order FEM (particularly with orders higher than 6) perform best on the Intel Xeon Phi with the MKL, while for solving the resulting linear system, PARDISO 6.0 with OpenBLAS on the

AMD EPYC is the best combination. The ratio between the solution system time and the assembly time gets higher as the polynomial order increases, as is expected with our efficient assembly algorithm, which improves on the results [61, 68] where the assembly time could still overshadow the solution time for high orders. The memory usage of direct solvers remains the real problem that fundamentally limits the size of the Helmholtz problems that can be tackled, and which, even more so than computing times, motivates the use of the domain decomposition methods.

Non-overlapping optimized Schwarz methods

2

As described in the previous chapter, direct solvers don't scale well for time-harmonic wave propagation problems in the high-frequency regime, both from a computational time and from a memory usage point of view. To overcome these limitations, in this chapter we introduce domain decomposition (DD) methods, which are hybrid solvers that couple direct and iterative solvers. First, we introduce traditional Schwarz methods and optimized Schwarz methods, and then we focus on the latter with both layered and block (checkerboard- and Rubik's-cube-type) decompositions of the mesh. The basic idea of domain decomposition techniques is to apply direct solvers on each subdomain independently, then exchange information between adjacent subdomains and iterate to recover the global solution. For block partitions, there are interior cross-points where more than two subdomains meet, and boundary cross-points that belong to both the exterior boundary and the subdomain interfaces, which require careful treatments. This chapter ends with a performance analysis and a discussion of the advantages and disadvantages of optimized Schwarz methods with these block partitions.

2.1 Schwarz methods

The earliest DD methods were introduced by H.A. Schwarz [81] in 1870, as a mathematical tool for solving partial differential equations in geometrical domains for which an explicit series solution was not known. A century later, P.-L. Lions [67] revisited this tool as a parallel computing strategy and proved fundamental convergence results.

The classical Schwarz method can be introduced by using the following example: consider Laplace's equation on a bounded Lipschitz domain Ω, in two or three dimensions, with Dirichlet boundary conditions. Suppose also that the original domain Ω is composed by two overlapping subdomains, Ω_I and Ω_J:

$$\overline{\Omega} = \overline{\Omega_I \cup \Omega_J}, \quad \Omega_I \cap \Omega_J \neq \emptyset, \quad \Gamma_I := \partial\Omega_I \cap \Omega_J, \quad \Gamma_J := \partial\Omega_J \cap \Omega_I;$$

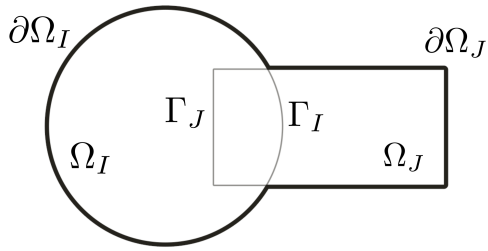


FIGURE 2.1: Overlapping domain decomposition configuration.

see Fig. 2.1. We also assume that the boundaries of the subdomains Ω_I, and Ω_J are Lipschitz continuous. The original problem is: Find $u(x, y)$ such that:

$$\begin{cases} \Delta u = 0 & \text{on } \Omega, \\ u = u_{\text{src}} & \text{on } \partial\Omega, \end{cases} \tag{2.1}$$

where $u_{\text{src}}(x, y)$ is a known function defined on the boundary of Ω, that is $\partial\Omega$. With such an overlapping domain decomposition configuration, the original problem can be solved by the following iterative scheme:

$$\begin{cases} \Delta u_I^{(n+1)} = 0 & \text{on } \Omega_I, \\ u_I^{(n+1)} = u_J^{(n)} & \text{on } \Gamma_I, \\ u_I^{(n+1)} = u_{\text{src}} & \text{on } \partial\Omega_I \setminus \Gamma_I, \\ \\ \Delta u_J^{(n+1)} = 0 & \text{on } \Omega_J, \\ u_J^{(n+1)} = u_I^{(n+1)} & \text{on } \Gamma_J, \\ u_J^{(n+1)} = u_{\text{src}} & \text{on } \partial\Omega_J \setminus \Gamma_J, \end{cases} \tag{2.2}$$

where $u_I^{(n)}$ and $u_J^{(n)}$ are the solution on Ω_I and Ω_J at iteration n, respectively. The boundaries Γ_I and Γ_J can also be defined as: $\Gamma_I = \partial\Omega_I \setminus \partial\Omega$, and $\Gamma_J = \partial\Omega_J \setminus \partial\Omega$. In (2.2), the algorithm alternatively solves Laplace's equation on subdomains Ω_I and Ω_J, hence its name: the alternating Schwarz method. See [67] for a proof that this method converges to the solution of the original problem.

It is obvious that the alternating algorithm (2.2) is sequential. It is however possible to improve this sequential version into a parallel version with a

simple "trick", by replacing the boundary value $u_I^{(n+1)}$ on Γ_J with $p_I^{(n)}$, leading to the so-called additive Schwarz method:

$$\begin{cases} \Delta u_I^{(n+1)} = 0 & \text{on } \Omega_I, \\ u_I^{(n+1)} = u_J^{(n)} & \text{on } \Gamma_I, \\ u_I^{(n+1)} = u_{\text{src}} & \text{on } \partial\Omega_I \setminus \Gamma_I, \\ \\ \Delta u_J^{(n+1)} = 0 & \text{on } \Omega_J, \\ u_J^{(n+1)} = u_I^{(n)} & \text{on } \Gamma_J, \\ u_J^{(n+1)} = u_{\text{src}} & \text{on } \partial\Omega_J \setminus \Gamma_J, \end{cases} \tag{2.3}$$

In this case, $u_I^{(n+1)}$ and $u_J^{(n+1)}$ can be independently calculated. P.-L. Lions [67] again proved that this parallel version of the Schwarz method converges to the solution of the original problem.

Next, we present non-overlapping Schwarz methods with two subdomains. The difference compared to overlapping Schwarz methods is that Ω is decomposed by two non-overlapping subdomains, Ω_I and Ω_J:

$$\overline{\Omega} = \overline{\Omega_I \cup \Omega_J}, \quad \Omega_I \cap \Omega_J = \varnothing;$$

see Fig. 2.2. Comparing to the case of an overlapping decomposition, the subproblems are not connected by Dirichlet conditions, but by Robin conditions. For the case in Figure 2.2, we have the following scheme:

$$\begin{cases} \Delta u_I^{(n+1)} = 0 & \text{on } \Omega_I, \\ (\mathbf{n}_{IJ} \cdot \mathbf{grad} + \lambda) u_I^{(n+1)} = (\mathbf{n}_{IJ} \cdot \mathbf{grad} + \lambda) u_J^{(n)} & \text{on } \Gamma_{IJ}, \\ u_I^{(n+1)} = u_{\text{src}} & \text{on } \partial\Omega_I \setminus \Gamma_{IJ}, \\ \\ \Delta u_J^{(n+1)} = 0 & \text{on } \Omega_J, \\ (\mathbf{n}_{JI} \cdot \mathbf{grad} + \lambda) u_J^{(n+1)} = (\mathbf{n}_{JI} \cdot \mathbf{grad} + \lambda) u_I^{(n)} & \text{on } \Gamma_{JI}, \\ u_J^{(n+1)} = u_{\text{src}} & \text{on } \partial\Omega_J \setminus \Gamma_{JI}, \end{cases} \tag{2.4}$$

where $\mathbf{n}_{IJ}$ is the unit vector outwardly normal to Ω_I, $\mathbf{n}_{JI}$ is the unit vector outwardly normal to Ω_J ($\mathbf{n}_{IJ} = -\mathbf{n}_{JI}$), and $\lambda \in \mathbb{R}_0^+$. The boundaries Γ_{IJ} and Γ_{JI} are now shared interfaces, which are defined as:

$$\Gamma_{IJ} = \Gamma_{JI} := \partial\Omega_I \cap \partial\Omega_J.$$

In (2.4), the 2nd equation and 5th equation are referred to as the transmission conditions for the non-overlapping domain decomposition method, and

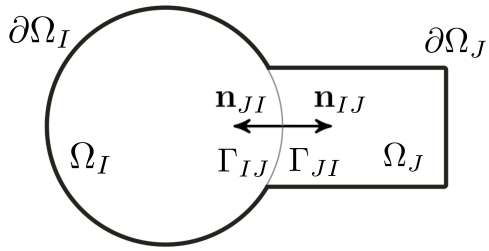


FIGURE 2.2: Non-overlapping domain decomposition configuration.

are also equivalent to the equality of any two independent linear combinations of the traces of the functions and their normal derivatives. This method can again be proved to converge to the solution of the original problem for Laplace's equation [67].

2.2 Optimized Schwarz methods for Helmholtz problems

In 1991, non-overlapping Schwarz methods were also proved to converge for the Helmholtz equation by Després [27], provided that one takes $\lambda = -ik$. This transmission condition at the interface between the two subdomains, which can be interpreted physically as an impedance transmission condition. The convergence of the resulting algorithm is however rather slow, which has led to considerable effort in designing improved transmission conditions, motivated by the fact that the optimal transmission condition, which would lead to convergence without iterations, would be to replace λ by the Dirichlet-to-Neumann (DtN) operator of the complementary subdomains. Since computing the exact DtN is not feasible in practice (as the related computational cost would be equivalent to solving the original problem), one focuses instead of finding generalized Robin operators $\mathcal{S}$, often defined in terms of some parameters to be optimized, leading to so-called optimized Schwarz methods. Such generalized operators $\mathcal{S}$ should be cheap to compute, but provide good approximations of the exact DtN.

Let the domain Ω be a bounded, smooth open set of $\mathbb{R}^d$, $d = 2$ or 3. We consider a decomposition of Ω into an arbitrary number N_{dom} of nonoverlapping subdomains $\Omega_1, \Omega_2, \ldots, \Omega_{N_{\text{dom}}}$ i.e. we assume that

$$\Omega = \Omega_1 \cup \cdots \cup \Omega_{N_{\text{dom}}} \cup \Gamma, \quad \Gamma = \bigcup_{I,J} \Gamma_{IJ},$$

and we define the set $D_I := \left\{J \in \{1, \dots, N_{\text{dom}}\} \text{ such that } J \neq I \text{ and } \Gamma_{IJ} \neq \varnothing\right\}$. With these definitions, we write our final version of the optimized Schwarz methods for Helmholtz problems as follows:

$$\begin{cases} (\Delta + k^2) u_I^{(n+1)} = 0 & \text{on } \Omega_I, \\ (\mathbf{n}_{IJ} \cdot \mathbf{grad} + \mathcal{S}) u_I^{(n+1)} = (\mathbf{n}_{IJ} \cdot \mathbf{grad} + \mathcal{S}) u_J^{(n)} & \text{on } \Gamma_{IJ}, \forall J \in D_I, \\ u_I^{(n+1)} = u_{\text{src}} & \text{on } \Gamma_{\text{src}} \cap \partial\Omega_I, \end{cases} \tag{2.5}$$

$\forall I \in \{1, 2, \dots, N_{\text{dom}}\}$, with Γ_{src} the boundary on which Dirichlet boundary conditions are prescribed. At the $(n+1)$-th iteration, we enforce boundary values of Ω_I being equal boundary values at n-th iteration of its adjacent subdomains and then solve all subproblems independently.

As the generalized Robin transmission operator $\mathcal{S}$ represents an approximation of the DtN operator on the interfaces, the better the approximation the faster the convergence rate of the optimized Schwarz methods. The first transmission operator, denoted by IBC(0), is the zeroth-order approximation of the DtN operator from Després:

$$\mathcal{S}^0 u = -iku.$$

This choice of the operator $\mathcal{S}$ leads to an algebraic convergence for optimized Schwarz method, which is often not satisfactory. A second choice for the transmission operator consists in second-order polynomial approximations of the DtN operator in Fourier space, with optimized coefficients [40]:

$$\mathcal{S}^{\text{oo2}} u = -(a + b\nabla_\Gamma \cdot \nabla_\Gamma) u.$$

The best choice of parameters a and b can be found in [40] for planar interfaces. The resulting transmission conditions are dubbed OO2. The third choice is the evanescent modes damping algorithm (EMDA), which is denoted by IBC(χ):

$$\mathcal{S}^\chi u = (-ik + \chi) u,$$

which was introduced by Bounbendir in [11, 14] and where χ is a self-adjoint positive definite operator (usually simply a real-valued positive coefficient). The last operator is the exact transmission operator [3, 12] for the half-space:

$$\mathcal{S}^{\text{sq},\epsilon} u = -ik\sqrt{1 + \nabla_\Gamma \cdot \left(\nabla_\Gamma / k_\epsilon^2\right)} u,$$

where $k_\epsilon = k + i\epsilon$ is a complex wavenumber, with optimal choices for ϵ depending on the geometry of the interfaces ($\epsilon = 0$ on planar interfaces). This nonlocal operator is then localized using Padé approximants. In what follows we will refer to this operator with Padé approximants as Padé-type High-order Absorbing Boundary Conditions (Padé-type HABCs), and denote them

by GIBC(N_p), where N_p is the number of terms in the Padé approximation, as they fall into the realm of so-called Generalized Impedance Boundary Conditions [12].

2.3 Krylov acceleration

The non-overlapping optimized Schwarz method can be recast as the solution of a linear system, which can be solved by a Krylov subspace method. Indeed, let's consider the transmission condition in (2.5) between two neighbouring subdomains Ω_I and Ω_J on the interface $\Gamma_{IJ} = \Gamma_{JI}$:

$$\begin{cases} (\mathbf{n}_{IJ} \cdot \mathbf{grad} + \mathcal{S}) u_I^{(n+1)} = (+\mathbf{n}_{IJ} \cdot \mathbf{grad} + \mathcal{S}) u_J^{(n)}, \\ \qquad\qquad\qquad\qquad = (-\mathbf{n}_{JI} \cdot \mathbf{grad} + \mathcal{S}) u_J^{(n)}, \quad \text{on } \Gamma_{IJ}, \\ (\mathbf{n}_{JI} \cdot \mathbf{grad} + \mathcal{S}) u_J^{(n+1)} = (+\mathbf{n}_{JI} \cdot \mathbf{grad} + \mathcal{S}) u_I^{(n)}, \\ \qquad\qquad\qquad\qquad = (-\mathbf{n}_{IJ} \cdot \mathbf{grad} + \mathcal{S}) u_I^{(n)}, \quad \text{on } \Gamma_{JI}. \end{cases} \tag{2.6}$$

We introduce two new quantities $g_{IJ}^{(n)}$ and $g_{JI}^{(n)}$ which describe the information coming from the subproblem in Ω_J and Ω_I at the n-th iteration, respectively:

1. $g_{IJ}^{(n)}$, new variables on Γ_{IJ} that describe the data coming from the subdomain Ω_J;
2. $g_{JI}^{(n)}$, new variables on Γ_{JI} that describe the data coming from the subdomain Ω_I.

According to these two new definitions, we have:

$$\begin{cases} g_{IJ}^{(n+1)} := (-\mathbf{n}_{JI} \cdot \mathbf{grad} + \mathcal{S})\, u_J^{(n+1)} = (-\mathbf{n}_{JI} \cdot \mathbf{grad} - \mathcal{S} + 2\mathcal{S})\, u_J^{(n+1)}, \\ g_{JI}^{(n+1)} := (-\mathbf{n}_{IJ} \cdot \mathbf{grad} + \mathcal{S})\, u_I^{(n+1)} = (-\mathbf{n}_{IJ} \cdot \mathbf{grad} - \mathcal{S} + 2\mathcal{S})\, u_I^{(n+1)}, \end{cases}$$

Substituting (2.6) into the above equation, we obtain update equations on Γ_{IJ} and Γ_{JI}, respectively:

$$\begin{cases} g_{IJ}^{(n+1)} = -g_{JI}^{(n)} + 2\mathcal{S}(u_J^{(n+1)}), \quad \text{on } \Gamma_{IJ}, \\ g_{JI}^{(n+1)} = -g_{IJ}^{(n)} + 2\mathcal{S}(u_I^{(n+1)}), \quad \text{on } \Gamma_{JI}, \end{cases}$$

With these definitions, the optimized Schwarz method (2.5) can be rewritten as follow:

$$\begin{cases} (\Delta + k^2) u_I^{(n+1)} = 0 & \text{on } \Omega_I, \\ (\mathbf{n}_{IJ} \cdot \mathbf{grad} + \mathcal{S}) u_I^{(n+1)} = g_{IJ}^{(n)} & \text{on } \Gamma_{IJ}, \forall J \in D_I, \\ u_I^{(n+1)} = u_{\text{src}} & \text{on } \Gamma_{\text{src}} \cap \partial\Omega_I, \end{cases} \tag{2.7}$$

with the update law

$$g_{IJ}^{(n+1)} = -g_{JI}^{(n)} + 2\mathcal{S}\left(u_J^{(n+1)}\right), \qquad \text{on } \Gamma_{IJ}.$$

Using a substructuring approach, we define the global transmission problem: the update of all the transmission variables at $(n+1)$-th iteration ($g_{IJ}^{(n)}, \forall J \in D_I$) is recast as one application of the iteration operator $\mathcal{A}$ defined by

$$\mathbf{g}^{(n+1)} = \mathcal{A}\mathbf{g}^{(n)} + \mathbf{b}, \tag{2.8}$$

where $\mathbf{g}^{(n)}$ is the set of all transmission variables defined on the interface faces and $\mathbf{b}$ is given by the source term. This algorithm is a Jacobi scheme applied to the linear system

$$(\mathcal{I} - \mathcal{A})\mathbf{g} = \mathbf{b}, \tag{2.9}$$

where $\mathcal{I}$ is the identity operator. In order to accelerate the convergence of the procedure, this system can be solved with Krylov subspace iterative methods, such as the GMRES method.

2.4 Optimized Schwarz methods with layered decompositions

We start by considering layered decompositions, in which an interface is shared by no more than two neighbors. Such decompositions avoid cross points that need special treatments, which will be introduced in the next section.

Considering that the initial computational domain Ω is split into N_{dom} non-overlapping subdomains Ω_i, $i = 1, \ldots, N_{\text{dom}}$, layered decompositions must satisfy:

- $\overline{\Omega} = \bigcup_{I=1}^{N_{\text{dom}}} \overline{\Omega}_I$,
- $\overline{\Omega}_I \bigcap \overline{\Omega}_J = \emptyset$, if $I \neq J$,
- $\partial\Omega_I \bigcap \partial\Omega_J = \Gamma_{IJ} = \Gamma_{JI}$ is the artificial interface separating Ω_I and Ω_J as long as its interior $\Gamma_{IJ} = \Gamma_{JI}$ is not empty.
- For all distince $I, J, K \in \{1, 2, \ldots, N_{\text{dom}}\}$, if $\Omega_I \bigcap \Omega_J \neq \emptyset$ and $\Omega_I \bigcap \Omega_K \neq \emptyset$, then $\Omega_J \bigcap \Omega_K = \emptyset$.

With these assumptions, the non-overlapping Schwarz algorithm with layered decompositions writes:

$$\begin{cases} (\Delta + k^2)u_I^{(n+1)} = 0 & \text{on } \Omega_I, \\ (\mathbf{n}_{IJ} \cdot \mathbf{grad} + \mathcal{S})u_I^{(n+1)} = (\mathbf{n}_{JI} \cdot \mathbf{grad} + \mathcal{S})u_J^{(n)} & \text{on } \Gamma_{IJ},\ \forall J \in D_I, \\ u_I^{(n+1)} = u_{\text{src}} & \text{on } \partial\Omega_I \cap \Gamma_{\text{src}}. \end{cases} \tag{2.10}$$

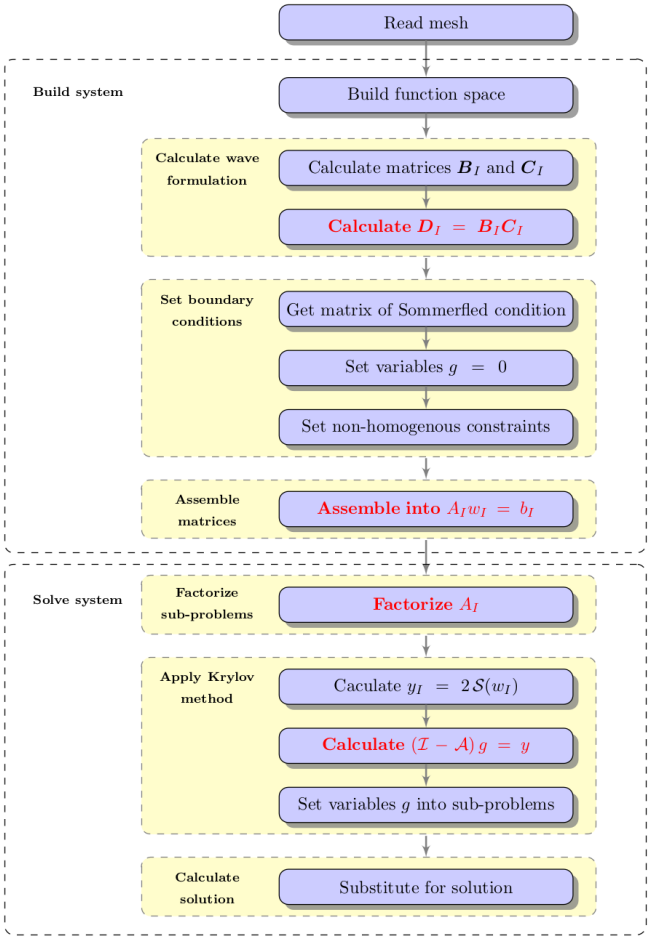


FIGURE 2.3: Numerical procedure of DD methods.

2.4.1 Performance analysis

In this section, the efficiency of DD methods is analyzed on the same architectures as in Chapter 1, with an emphasis on the on AMD EPYC 7551 since it has the best parallel performance overall.

Figure 2.3 shows the general numerical procedure of Helmholtz solvers combined with DD methods. It is similar to the flowchart for direct solvers (cf. Figure 1.3), with the same two main phases ("Build system" and "Solve system"), but with additional steps in each phase corresponding to the steps of the DD method, with additional constraints on the artificial interfaces and the presence of the iterative GMRES method. In all the examples below the relative tolerance for the GMRES method is set to 10^{-5}.

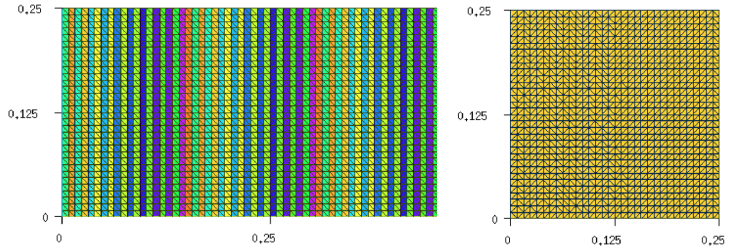


FIGURE 2.4: Example of mesh used for DD methods with layered decomposition.

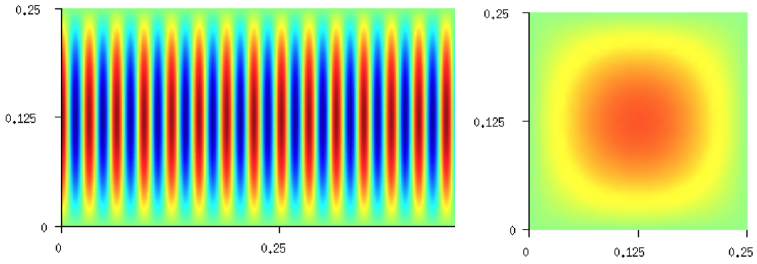


FIGURE 2.5: Example of numerical solution of DD methods with layered decomposition, with $k = 200$.

2.4.1.1 Mesh and memory usage

We consider a three-dimensional acoustic waveguide of size 0.5 m with a cross section of 0.25×0.25 m^2. The geometry is meshed with 32 tetrahedra per wavelength, and the default wavenumber is set to 25 rad/m. The global domain is sliced with a layered decomposition, with 64 subdomains by default, with each subdomain attributed to a single core. Different wavenumbers and number of subdomains will be considered as well to highlight the efficiency of the code. The transmission conditions on the interfaces are the second-order transmission conditions, and polynomial orders from 1 to 7 are again considered.

The size of the DD methods' system is slightly larger than that of the direct solvers' system, due to the additional unknowns on the artificial interfaces between adjacent subdomains. Despite this slight increase in the number of unknowns, the memory usage of DD methods dramatically decreases compared to that of direct solvers, thanks to the smaller matrices on which factorizations are computed: see Table 2.1.

#Tetrahedra	Order	System size (million)		Memory usage (GB)	
		Direct solver	DD method	Direct solver	DD method
393216	1	0.1	0.1	1.1	0.2
	2	0.5	0.8	15.9	2.9
	3	1.7	2.3	81.5	13.4
	4	4.0	5.1	272.3	42.9
	5	7.9	9.7	693.6	108.8
	6	13.8	16.3	1487.6	244.5
	7	24.3	28.5	4027.3	600.9

TABLE 2.1: Information on mesh, global system size and memory usage (64 subdomains).

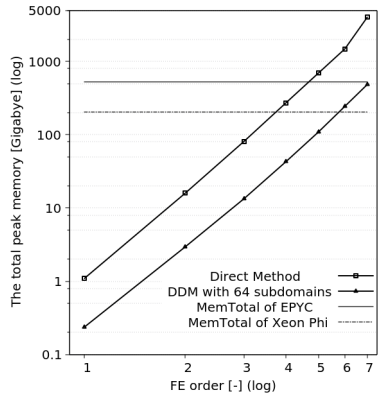


FIGURE 2.6: Memory usage of direct solvers and DD methods.

2.4.1.2 Benchmarks

Figure 2.7 compares the performance of the matrix-matrix products and of matrix assembly between the direct solver approach from the previous chapter and the proposed DD method with layered decomposition. The higher efficiency of the DD method is due to the fact that one core is pinned for each subdomain, which avoids any synchronization.

Figure 2.8 reports the computation times of matrix factorization for the DD approach for different polynomial orders, in function of the number of subdomains (which is equal to the number of threads, as we still use one thread per subdomain). The factorization time decreases dramatically with the number of subdomains, with parallel efficiency increasing with the polynomial order.

In order to test the ability the overall potential of DD methods, Table 2.2 reports the L2-error and the number of GMRES iterations in terms of the poly-

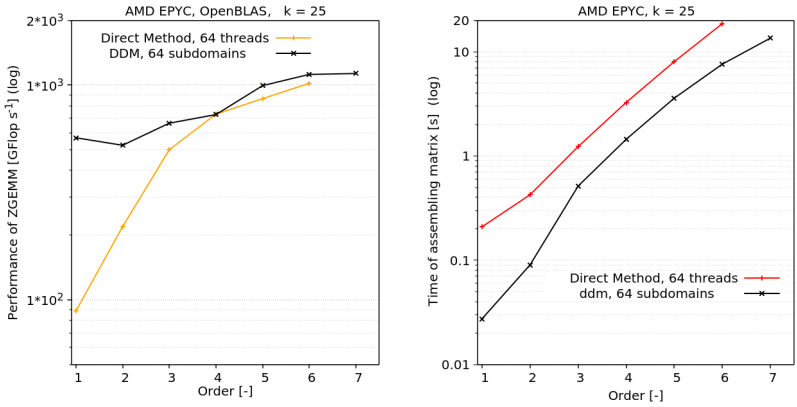


FIGURE 2.7: Comparison of matrix-matrix product and assembly performance on the AMD EPYC, with a direct solver (Direct Method) and the DD method (DDM) with one thread per subdomain.

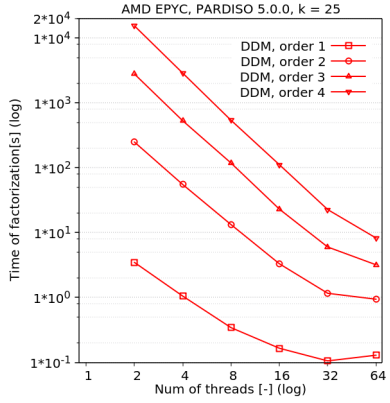


FIGURE 2.8: Factorization time for the DD method on EPYC, with one thread per subdomain.

nomial order and the wavenumber. The interest of using higher finite element orders is clearly visible, with stable iteration counts w.r.t. to frequency. No results are available at order 7 because this test-case exceeded the total memory available on the EPYC.

Finally, the total computation time of the DD method is compared to the computation time of the direct solver approach in Figure 2.9. While the direct solver approach possesses a slight edge for linear finite elements, the DD approach vastly outperforms the best direct solver on both the AMD Epyc and the Intel Xeon Phi for higher orders. The direct solver approach on the Xeon Phi actually is not able to handle polynomial orders higher than 3, due

Order	1	2	3	4	5	6
Wavenumber	25	50	75	100	150	200
L2 Error	2.9E-02	3.9E-04	7.8E-05	8.7E-06	2.6E-06	1.7E-06
Num. of iter.	139	146	130	126	126	126

TABLE 2.2: L2 error and number of iterations with increasing wavenumber and polynomial order.

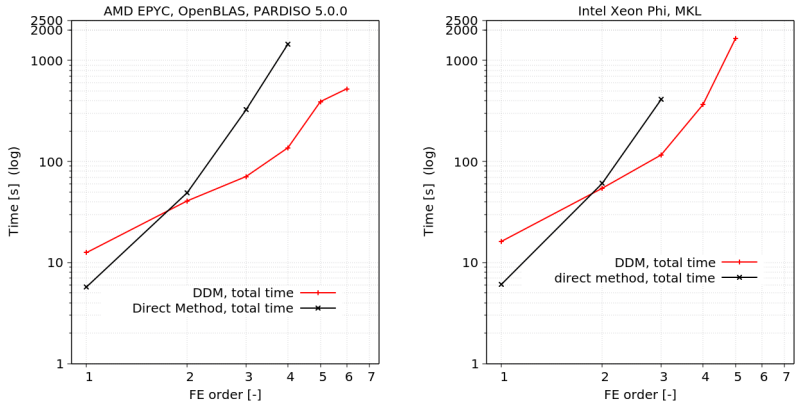


FIGURE 2.9: Total computational times on the AMD EPYC with a direct solver (Direct Method) and the DD method (DDM), with one thread per subdomain.

to excessive memory usage because of excess fill-in.

In order to analyze the performance of DD methods more finely, Figure 2.10 breaks down the computation times of DD methods for polynomial orders 1, 3 and 5 on the AMD Epyc, in function of the number of subdomains. The time spent during Krylov iterations increases with the number of subdomains, which is expected for an unpreconditioned, one-level DD method. This is verified on Figure 2.11, where the number of GMRES iterations is reported in terms of the number of subdomains, the polynomial order and the wavenumber.

In order to improve the overall computational times for small number of subdomains, let us know always fully utilize the total number of threads (64) available on the AMD Epyc, i.e. by using as many threads as possible for the forward and backward substitutions using nested parallel regions. Figure 2.12 reports the timings of the DD method within such nested parallel regions (to be compared with Figure 2.10, with one thread per subdomain).

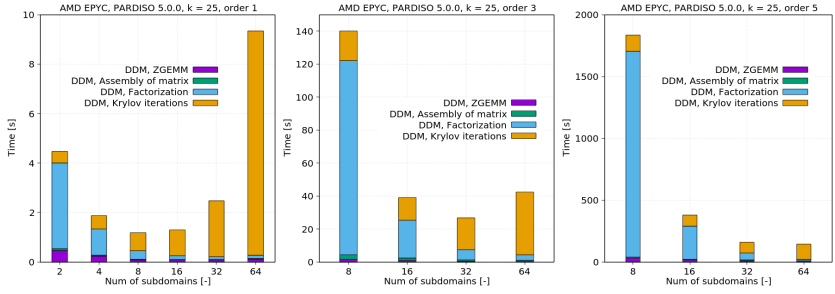
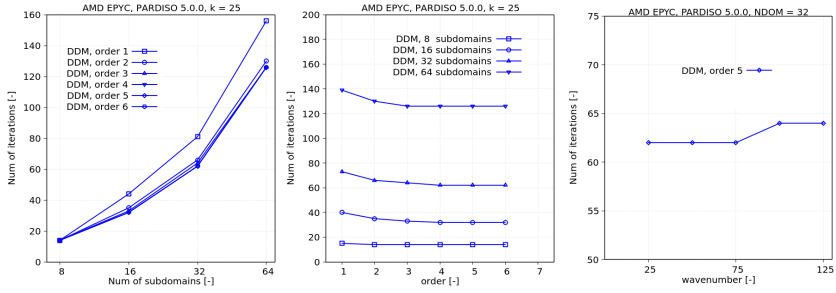


FIGURE 2.10: Breakdown of DD method computation time on the AMD EPYC, with one thread per subdomain.



(A) Num. of iter. vs. N_{dom} (B) Num. of iter. vs. p. order (C) Num. of iter. vs. k

FIGURE 2.11: Convergence for waveguide decomposition.

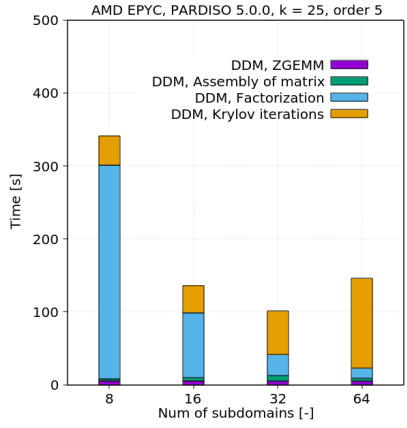


FIGURE 2.12: Breakdown of DD method computation time on the AMD EPYC, with nested parallel regions (64 threads for all computations).

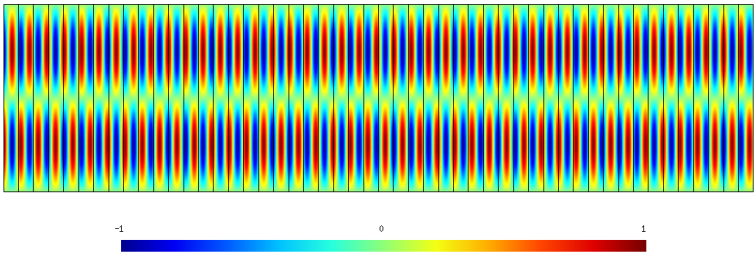


FIGURE 2.13: Homogeneous waveguide ($D = 4; d = 1$) decomposition and solution for $\omega = 20\pi$.

	$\omega = 20\pi$				
	$N_{\text{dom}} = 5$	10	25	50	100
IBC(0)	8	18	48	98	198
IBC($k/2$)	8	18	48	112	306
OO2	8	18	48	98	198
GIBC(2)	8	18	46	98	202
GIBC(8)	8	18	48	98	198

TABLE 2.3: Homogeneous waveguide: number of GMRES iterations (without restart) to reach a relative residual of 10^{-6}, as a function of the number of subdomains N_{dom} in a layered decomposition, for different transmission conditions.

The overall computation times for number of subdomains smaller than 64 is clear improved, and actually the optimal choice for this particular test case on the AMD Epyc should be 32 subdomains.

2.4.2 Choice of the transmission conditions

The analysis performed in the previous section shows that Schwarz DD methods with second-order transmission conditions are a viable method to handle high-frequency Helmholtz problems on modern multicore architectures like the Intel Xeon Phi and the AMD Epyc. In this section we replicate the simulations from [90] with our implementation to gauge the interest of using other transmission conditions. As introduced in Section 2.2, besides the optimized second order condition OO2, we consider the zeroth-order impedance condition IBC(0) and the evanescent mode damping condition IBC(χ), as well as the generalized impedance boundary conditions with Padé localization GIBC(N_p).

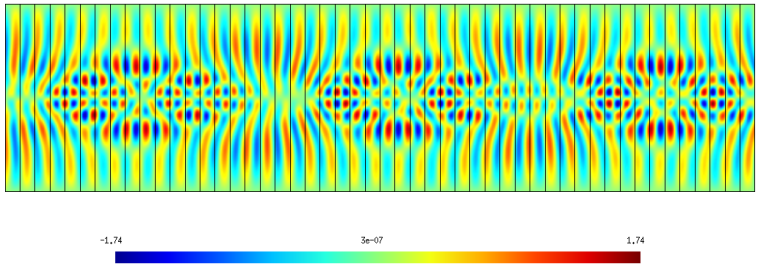


FIGURE 2.14: Heterogeneous waveguide ($D = 4; d = 1$) decomposition and solution for $\omega = 20\pi$.

	$\omega = 20\pi$				
	$N_{\text{dom}} = 5$	10	25	50	100
IBC(0)	55	106	299	dnc	dnc
IBC($k/2$)	59	126	358	dnc	dnc
OO2	53	105	314	dnc	dnc
GIBC(2)	38	82	222	430	dnc
GIBC(8)	38	83	217	413	dnc

TABLE 2.4: Non-homogeneous waveguide: number of GMRES iterations (without restart) to reach a relative residual of 10^{-6}, as a function of the number of subdomains N_{dom} in a layered decomposition, for different transmission conditions. The abbreviation "dnc" stands for "did not converge" within the prescribed maximum 500 iterations.

The first test case is a homogeneous waveguide with a rectangle geometry ($\Omega = [0, D] \times [0, d]$). Homogeneous Dirichlet conditions are prescribed on the upper and lower sides of the rectangle: $u = 0$ on $y = \{0, d\}$. The second waveguide mode is excited on the left side of the rectangle:

$$u(0, y) = \sin(m\pi y\, d),$$

with $m = 2$, and a Sommerfeld absorbing condition is imposed on the right of the rectangle. In this simple case all transmission conditions perform equally well. And as expected the number of iterations increases (almost) linearly with the number of subdomains.

The second test case is similar to the previous one, but with a velocity profile that is constant in the propagation direction (horizontal direction) and Gaussian in the transverse direction (vertical direction):

$$c(x, y) = 1.25\,(1 - 0.4\, e^{-32\,(y-0.5)^2}).$$

The source and boundary conditions are the same as in the first case. The heterogeneity makes the simulation truly two-dimensional, with a much more complex wave propagation, and highlights the effectiveness of the improved high-order Padé-type High-order Absorbing Boundary Conditions GIBC in more complex configurations.

The increase of the number of GMRES iterations with the number of subdomains is still clearly visible though, which motivates the use of non-layered decompositions, as presented in the next sections, coupled with a preconditioning strategy–which will be proposed in the next chapter.

2.5 Optimized Schwarz methods with block decompositions

Block decompositions, also sometimes called "checkerboard decompositions", "lattice-type decompositions" or "Rubik's-cube decompositions", consist of subdomains which are rectangles in two dimensions, or cubes in three dimensions. For block decompositions, there may be some points or edges where more than two subdomains meet, which are called "cross-points" or "cross-edges". It is worth noting that there are *interior* cross-points and cross-edges where more than two subdomains meet, and *boundary* cross-points and cross-edges which belong to both the exterior boundary and the artificial interfaces. For clarity from now on we will simply use the term "cross-points" to denote either cross-points or cross-edges.

For zeroth-order transmission condition IBC(0) there is no additional requirement in the presence of cross-points to analyze optimized Schwarz methods at the continuous level. However, there is an issue at the discrete level, especially in the case where nodal discretizations are used. For elliptic problems, it is pointed out in [39] that nodal discretization for which degrees of freedom are associated with cross-points may diverge and that the energy estimate for the continuous setting fails for the discrete setting. A solution is proposed in [13, 6], which however leads to a global system that needs to be solved at each iteration. While the size of the system is moderate (equal to the number of cross-points), it does not mesh well with a distributed DD approach. Moreover, no matter what setting one has, the zeroth-order transmission operator is not a very good approximation of the DtN operator, which limits the convergence rate of the DD method.

For the second-order transmission conditions, there are some difficulties in the cross-points at the continuous level. One needs some additional conditions at the cross-points to ensure the well-posedness of the problems with the second-order transmission conditions. In [54], the well-posedness of the solution to the Helmholtz equation with second-order transmission conditions in a rectangle is proved. This problem is completed by compatibility conditions at the corner.

For Padé-type high-order transmission conditions, there are some second-order auxiliary equations, which lead to the same difficulty at the corner at a rectangle as with the second-order transmission conditions. Modave *et al* [70] proposed practical compatibility conditions in this context and applied them to optimized Schwarz methods for Helmholtz problems in [71].

While conditions based on second-order operators [76], perfectly matched layers (PML) [84, 91, 4] and non-local approaches [62, 23] could also been investigated, we chose to follow the approach developed in [70, 71] as it is a direct extension of the conditions we successfully implemented and tested in Section 2.4.2. Moreover, the compatibility conditions are exact in the case of right angles, which makes them a perfect fit for block decompositions.

2.5.1 Optimized Schwarz methods on a checkerboard partition

We consider a checkerboard partition of the domain Ω, that consists in a lattice of rectangular non-overlapping subdomains Ω_I ($I = 1 \dots N_{\text{dom}}$) with N_r rows and N_c columns (then, $N_{\text{dom}} = N_r \times N_c$). For each rectangular subdomain Ω_I, there are four edges, which are on the left, on the bottom, on the right, and on the top of the subdomain, respectively (see Figure 2.15, right). and we define the set

$$D_I^\infty := \left\{ J \in \{-1, -2, -3, -4\} \text{ such that } \Gamma_{IJ} \neq \varnothing \right\},$$

where the superscripts -1, -2, -3, and -4 correspond to the non-empty edges belonging to $\partial\Omega$ on the left, on the bottom, on the right, and on the top of the subdomain, respectively. The union of the edges of Ω_I then reads

$$\left(\bigcup_{J \in D_I} \Gamma_{IJ}\right) \bigcup \left(\bigcup_{J \in D_I^\infty} \Gamma_{IJ}\right).$$

To simplify the presentation, we assume that the obstacle is included in only one subdomain, the boundary of which is the union of the four edges and the boundary of the obstacle (see Figure 2.15, middle).

Each edge of one subdomain Ω_I is either a *boundary edge* if it belongs to the boundary of the global domain ($J \in D_I^\infty$) or an *interface edge* if there is a neighboring subdomain on the other side of the edge ($J \in D_I$). In this checkerboard partition, there are corners where at least two edges meet. Each corner of a subdomain is an *interior cross-point* (point that belongs to four subdomains), a *boundary cross-point* (point that belongs to two subdomains and to the exterior border $\partial\Omega$) or a corner of the main domain Ω.

With these definitions, the non-overlapping domain decomposition algorithm can be set up as follows. For each subdomain Ω_I, we seek the solution

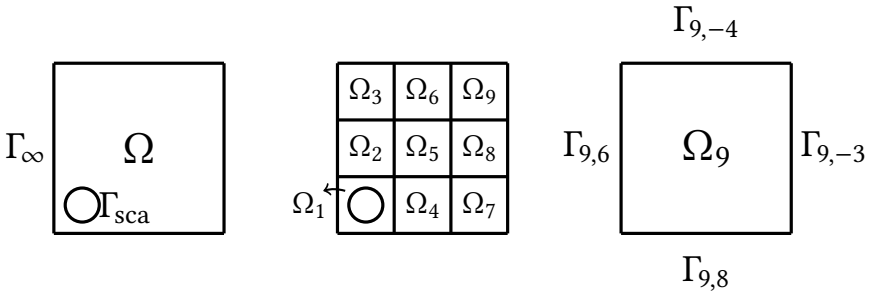


FIGURE 2.15: Configuration of the problem *(left)*, illustration of the checkerboard partition *(middle)* and notation for the edges of the subdomain Ω_I *(right)*.

$u_I(\mathbf{x})$ of the subproblem

$$\begin{cases} -\Delta u_I - \kappa^2 u_I = 0, & \text{in } \Omega_I, \\ \partial_{\boldsymbol{n}_{IJ}} u_I + \mathcal{B}_{IJ}\, u_I = 0, & \text{on each } \Gamma_{IJ}, \forall J \in D_I^\infty, \\ \partial_{\boldsymbol{n}_{IJ}} u_I + \mathcal{B}_{IJ}\, u_I = g_{IJ}, & \text{on each } \Gamma_{IJ}, \forall J \in D_I, \\ u_I = -u_{\text{inc}}, & \text{on } \partial\Omega_I \cap \Gamma_{\text{sca}}, \end{cases} \tag{2.11}$$

where $\boldsymbol{n}_{IJ}$ is the outward unit normal to the edge Γ_{IJ}, $\mathcal{B}_{IJ}$ is an impedance operator and g_{IJ} is a transmission variable defined on Γ_{IJ}. The second and third equations in (2.11) are boundary and transmission conditions, respectively.

For a given boundary edge $\Gamma_{IJ} \subset \partial\Omega$, we must have $\mathcal{B}_{IJ} = \mathcal{B}$ to ensure the equivalence between all the subproblems and the original problem. If $\Gamma_{IJ} \not\subset \partial\Omega$, there is some flexibility in the choice of $\mathcal{B}_{IJ}$. The transmission variable is defined as

$$g_{IJ} := \partial_{\boldsymbol{n}_{IJ}} u_J + \mathcal{B}_{IJ}\, u_J, \tag{2.12}$$

where u_J is the solution of the neighboring subdomain Ω_J. The transmission conditions defined on both sides of the interface enforce the continuity of the solution across the interface. Assuming that the impedance operators used on both sides of the shared interface edge $\Gamma_{IJ} = \Gamma_{Ji} = \partial\Omega_I \cap \partial\Omega_J$ are the same (*i.e.* $\mathcal{B}_{IJ} = \mathcal{B}_{JI}$), the transmission variables defined on this edge verify

$$g_{IJ} = -g_{JI} + 2\mathcal{B}_{JI}\, u_J, \tag{2.13}$$

where g_{JI} is the transmission variable defined on the edge Γ_{JI} of Ω_J.

The non-overlapping optimized Schwarz domain decomposition algorithm consists in solving subproblems associated to all the subdomains (equation (2.11)) concurrently and updating the transmissions variables using (2.13) in an interative process. At each iteration $n+1$, the update formula of a transmission variable living on an interface edge Γ_{IJ} of a subdomain Ω_I reads

$$g_{IJ}^{(n+1)} = -g_{JI}^{(n)} + 2\mathcal{B}_{JI}\, u_J^{(n)}, \tag{2.14}$$

where $u_J^{(n)}$ is the solution of the neighboring subdomain Ω_J at the iteration n. The update of all the transmission variables can be recast as one application of the iteration operator $\mathcal{A}$ defined by

$$\mathbf{g}^{(n+1)} = \mathcal{A}\mathbf{g}^{(n)} + \mathbf{b}, \tag{2.15}$$

where $\mathbf{g}^{(n)}$ is the set of all transmission variables defined on the interface edges and $\mathbf{b}$ is given by the source term. It is well known that this algorithm can be seen as a Jacobi scheme applied to the linear system

$$(\mathcal{I} - \mathcal{A})\mathbf{g} = \mathbf{b}, \tag{2.16}$$

where $\mathcal{I}$ is the identity operator. In order to accelerate the convergence of the procedure, this resulting system can be solved with the Krylov subspace iterative methods, such as GMRES, etc.

2.5.2 High-order transmission operators

Let us now detail how the chosen Padé-type HABC generalized impedance transmission conditions can be adapted for configurations with cross-points, following [70, 71]. For an edge Γ_{IJ}, the operator can be written as

$$\mathcal{B}_{IJ} = -\imath\kappa\alpha \left[1 + \frac{2}{M} \sum_{i=1}^{N} c_i \left(1 - \alpha^2 (c_i + 1) \left[(\alpha^2 c_i + 1) + \partial_{\boldsymbol{\tau}\boldsymbol{\tau}} / \kappa^2 \right]^{-1} \right) \right], \quad \text{on } \Gamma_{IJ}, \tag{2.17}$$

where $\partial_{\boldsymbol{\tau}}$ is the tangential derivative and we have $\alpha = e^{\imath\phi/2}$, $c_i = \tan^2(i\pi/M)$ and $M = 2N + 1$. This Padé-type impedance operator is obtained by approximating the exact non-local DtN map associated to the exterior half-plane problem (see *e.g.* [30, 69]). The symbol of the non-local operator exhibits a square-root which is replaced with the $(2N+1)^{\text{th}}$-order Padé approximation after a ϕ-rotation of the branch-cut. The performance of the obtained operator depends on the number of terms N and the angle of rotation ϕ. The particular parameters $N = 0$ and $\phi = 0$ leads to the basic ABC operator $\mathcal{B}_{IJ} = -\imath\kappa$. See *e.g.* [56, 70] for further details.

For the effective implementation of the transmission condition, the application of the Padé-type impedance operator on a field is written in such a way that it involves only differential operators. Following an approach first used by Lindman [65] for ABCs, we introduce N auxiliary fields governed by auxiliary equations on interface edge Γ_{IJ}. The application of the Padé-type impedance operator is then written as

$$\mathcal{B}_{IJ} u_I = B\Big(u_I, \{\varphi_{IJ,i}\}_{i=1\ldots N}\Big) := -\imath\kappa\alpha \left[u_I + \frac{2}{M} \sum_{i=1}^{N} c_i \left(u_I + \varphi_{IJ,i} \right) \right], \quad \text{on } \Gamma_{IJ}, \tag{2.18}$$

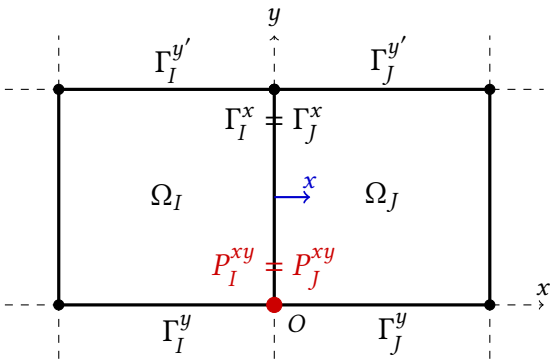


FIGURE 2.16: Configuration with two subdomains.

with the auxiliary fields $\{\varphi_{IJ,i}\}_{i=1\dots N}$ defined only on the edge and governed by the auxiliary equations

$$-\partial_{\tau\tau}\varphi_{IJ,i} - \kappa^2\Big((\alpha^2 c_i + 1)\varphi_{IJ,i} + \alpha^2(c_i+1)u_I\Big) = 0, \quad \text{on } \Gamma_{IJ}, \tag{2.19}$$

for $i = 1\dots N$. The linear multivariate function B is introduced to simplify the expressions in the remainder of the paper. When this operator is used in a boundary condition for polygonal domains, a special treatment is required to preserve the accuracy of the solution at the corners. In the case of right-angle corners, an approach based on compatibility relations reveals to be very efficient [70].

2.5.3 Optimized Schwarz methods with compatibility relations in 2D

When the Padé-type impedance operator (2.18) is used in the boundary conditions and the interface conditions of the subproblem (2.11), a special treatment is required at the corners of the subdomain Ω_I. Indeed, the auxiliary fields governed by equation (2.19) on the edges require boundary conditions at the extremities of the edges, which are corners of the subdomain.

To present the cross-point treatment, we consider the neighboring subdomains Ω_I and Ω_J represented on Figure 2.16. For the sake of conciseness, we describe the methodology in the case where transmission conditions are prescribed on all the edges of both subdomains (*i.e.* they all are interface edges) and the Padé-type impedance operator is used with the same parameters on all the edges.

The subdomains share the interface edge $\Gamma_I^x = \Gamma_J^x$ ($\Gamma_I^x := \Gamma_{IJ}$ and $\Gamma_J^x := \Gamma_{JI}$) and the interior cross-points $P_I^{xy} = P_J^{xy}$ and $P_I^{xy'} = P_J^{xy'}$. On the shared

interface edge $\Gamma_I^x = \Gamma_J^x$, we have the transmission conditions

$$\partial_x u_I + B\Big(u_I, \{\varphi_{I,i}^x\}_{i=1\dots N}\Big) = g_I^x, \quad \text{on } \Gamma_I^x, \tag{2.20}$$

$$-\partial_x u_J + B\Big(u_J, \{\varphi_{J,i}^x\}_{i=1\dots N}\Big) = g_J^x, \quad \text{on } \Gamma_J^x, \tag{2.21}$$

where u_I and u_J are the main fields defined on Ω_I and Ω_J, respectively. The auxiliary fields $\{\varphi_{I,i}^x\}_i := \{\varphi_{IJ,i}\}_i$ and $\{\varphi_{J,i}^x\}_i := \{\varphi_{JI,i}\}_i$ defined on the shared interface are governed by equation (2.19). The first set of auxiliary fields is associated to the subproblem defined on Ω_I, and the second set is associated to the one defined on Ω_J. By using the impedance operator (2.18) in equation (2.13) on both sides of the interface, we have that the transmission variables g_I^x and g_J^x verify

$$g_I^x = -g_J^x + 2B\Big(u_J, \{\varphi_{J,i}^x\}_{i=1\dots N}\Big), \quad \text{on } \Gamma_I^x, \tag{2.22}$$

$$g_J^x = -g_I^x + 2B\Big(u_I, \{\varphi_{I,i}^x\}_{i=1\dots N}\Big), \quad \text{on } \Gamma_J^x. \tag{2.23}$$

With these transmission variables, the transmission conditions (2.20) and (2.21) enforce weakly the continuity of the main field across the shared interface.

The cross-point treatment consists in enforcing weakly the continuity of auxiliary fields at cross-points. More precisely, only auxiliary fields defined on edges that are aligned are continuous. For instance, the auxiliary fields $\{\varphi_{I,j}^y\}_j$ defined on Γ_I^y and the auxiliary fields $\{\varphi_{J,j}^y\}_j$ defined on Γ_J^y (*i.e.* defined on the upper edges of Ω_I and Ω_J, respectively, see Figure 2.16) must be equal at $P_I^{xy} = P_J^{xy}$. Following the approach detailed in [71], transmission conditions with specific impedance operators are used to enforce weakly the continuity. At the cross-point, we use the transmission conditions

$$\partial_x \varphi_{I,j}^y + B\Big(\varphi_{I,j}^y, \{\psi_{I,ij}^{xy}\}_{i=1\dots N}\Big) = g_{I,j}^{xy}, \quad \text{at } P_I^{xy}, \tag{2.24}$$

$$-\partial_x \varphi_{J,j}^y + B\Big(\varphi_{J,j}^y, \{\psi_{J,ij}^{xy}\}_{i=1\dots N}\Big) = g_{J,j}^{xy}, \quad \text{at } P_J^{xy}, \tag{2.25}$$

for $j = 1 \dots N$, with the scalar variables $\{\psi_{I,ij}^{xy}\}_{ij}$ and $\{\psi_{J,ij}^{xy}\}_{ij}$ defined as

$$\psi_{I,ij}^{xy} = -\Big[\alpha^2(c_j+1)\varphi_{I,i}^x + \alpha^2(c_i+1)\varphi_{I,j}^y\Big] \Big/ \Big[\alpha^2 c_i + \alpha^2 c_j + 1\Big], \quad \text{at } P_I^{xy}, \tag{2.26}$$

$$\psi_{J,ij}^{xy} = -\Big[\alpha^2(c_j+1)\varphi_{J,i}^x + \alpha^2(c_i+1)\varphi_{J,j}^y\Big] \Big/ \Big[\alpha^2 c_i + \alpha^2 c_j + 1\Big], \quad \text{at } P_J^{xy}, \tag{2.27}$$

for $i, j = 1 \dots N$. At the cross-point, the new transmission variables $\{g_{I,j}^x\}_j$ and $\{g_{J,j}^x\}_j$ verify

$$g_{I,j}^{xy} = -g_{J,j}^{xy} + 2B\Big(\varphi_{J,j}^y, \{\psi_{J,ij}^{xy}\}_{i=1\dots N}\Big), \quad \text{at } P_I^{xy}, \tag{2.28}$$

$$g_{J,j}^{xy} = -g_{I,j}^{xy} + 2B\Big(\varphi_{I,j}^y, \{\psi_{I,ij}^{xy}\}_{i=1\dots N}\Big), \quad \text{at } P_J^{xy}, \tag{2.29}$$

for $j = 1 \dots N$. In a nutshell, the same condition with the multivariate function B is used on the interface to couple the main fields (equations (2.20)-(2.21)) and at the cross-points to couple the auxiliary fields (equations (2.26)-(2.27)). The scalar variables defined at the corners of the subdomains introduce a coupling of auxiliary fields living on adjacent edges of each subdomain. This strategy can be adapted rather straightforwardly to deal with boundary cross-points, where interface edges and boundary edges with boundary condition meet. For further details, we refer to [71].

The iterative domain decomposition algorithm is very similar to the algorithm described at the end of section 2.5.1. At each iteration, subproblems associated to the subdomains are solved concurrently, and transmission variables are updated. Here, the subproblem associated to Ω_I consists in finding the main field verifying system (2.11) and auxiliary fields verifying equations similar to (2.19) on each interface edge. The transmission variables are associated to interfaces edges and cross-points. They are updated with formulas similar to

$$g_I^{x\,(n+1)} = -g_J^{x\,(n)} + 2B\Big(u_J^{(n)}, \{\varphi_{J,i}^{x\,(n)}\}_{i=1\dots N}\Big), \quad \text{on } \Gamma_I^x \tag{2.30}$$

and

$$g_{I,j}^{xy\,(n+1)} = -g_{J,j}^{xy\,(n)} + 2B\Big(\varphi_{J,i}^{y\,(n)}, \{\psi_{J,ij}^{xy\,(n)}\}_{i=1\dots N}\Big), \quad \text{on } P_I^{xy}, \tag{2.31}$$

which are obtained by rewritting equations (2.22) and (2.28). Here, g_{IJ} can be defined as

$$g_{IJ} \coloneqq \begin{bmatrix} g_I^x \\ g_{I,j}^{xy} \\ g_{I,j}^{xy'} \end{bmatrix}, \tag{2.32}$$

where $g_{I,j}^{xy'}$ is the transmission variable at $P_I^{xy'}$. One can consider equations (2.22) and (2.28), and the formula at $P_I^{xy'}$ that is the same to (2.28), which leads to the following formula:

$$g_{IJ}^{(n+1)} = -g_{JI}^{(n)} + \begin{bmatrix} 2B\Big(u_J^{(n)}, \{\varphi_{J,i}^{x\,(n)}\}_{i=1\dots N}\Big) \\ 2B\Big(\varphi_{J,i}^{y\,(n)}, \{\psi_{J,ij}^{xy'\,(n)}\}_{i=1\dots N}\Big) \\ 2B\Big(\varphi_{J,i}^{y'\,(n)}, \{\psi_{J,ij}^{xy'\,(n)}\}_{i=1\dots N}\Big) \end{bmatrix}. \tag{2.33}$$

This formula is similar to the general update formula (2.14). Following the approach explained in section 2.5.1, all the transmission variables can be merged into a global vector $\mathbf{g}^{(n+1)}$, and the global process can be recast as one application of an iterative operator $\mathcal{A}$ on the vector. At each iteration $n+1$, the

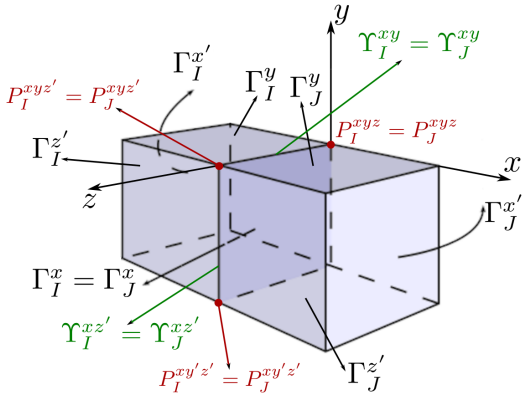


FIGURE 2.17: Illustration of neighbouring subdomains Ω_I and Ω_J in three dimensions.

whole process can be seen as one step of the Jacobi method to solve the linear system $(\mathcal{I} - \mathcal{A})\mathbf{g} = \mathbf{b}$, which could be solved with a Krylov subspace iterative method. Here, the main difference with most of the works is that the global vector includes transmission variables associated to both interfaces and cross-points.

2.5.4 Optimized Schwarz methods with compatibility relations in 3D

Next, we extend the approach in two dimensions proposed in [71] in three dimensions. Compared to the compatibility relations in two dimensions, compatibility relations in three dimensions must be prescribed at the edges and the corners of the cuboid. Let us consider the three-dimensional version of the Helmholtz equation, and two subdomains Ω_I and Ω_J in two regions $(-,-,-)$ and $(+,-,-)$ of a Euclidean three-dimensional coordinate system, respectively (see Figure 2.17). The shared faces of these two subdomains belonging to the place $x = 0$ are denoted $\Gamma_I^x := \Gamma_{IJ}$ and $\Gamma_J^x := \Gamma_{JI}$ ($\Gamma_{IJ} = \Gamma_{JI}$). The faces of these two subdomains belonging to the planes $y = 0$ and $z = 0$ are denoted Γ_I^y, Γ_J^y and Γ_I^z, Γ_J^z. For Ω_I, the faces that are parallel to Γ_I^x, Γ_I^y, and Γ_I^z are denoted $\Gamma_I^{x'}$, $\Gamma_I^{y'}$, and $\Gamma_I^{z'}$. For Ω_J, the parallel faces are denoted in the same way. The edges of Ω_I are given by $\Upsilon_I^{xy} = \overline{\Gamma_I^x} \cap \overline{\Gamma_I^y}$, $\Upsilon_I^{xz} = \overline{\Gamma_I^x} \cap \overline{\Gamma_I^z}$, $\Upsilon_I^{yz} = \overline{\Gamma_I^y} \cap \overline{\Gamma_I^z}$, etc. The corners of Ω_I are given by $P_I^{xyz} = \overline{\Gamma_I^x} \cap \overline{\Gamma_I^y} \cap \overline{\Gamma_I^z}$, etc. The edges and corners of Ω_J are denoted in the same way.

Similarily to the case in two dimensions, we describe the methodology in

three dimensions in the case where transmission conditions are prescribed on all the faces of these two subdomains and the Padé-type impedance operator is used with the same parameters on all the faces.

On the shared face $\Gamma_I^x = \Gamma_J^x$, we have the transmission conditions

$$\partial_x u_I + B\Big(u_I, \varphi_{I,1}^x, \ldots, \varphi_{I,N}^x\Big) = g_I^x, \quad \text{on } \Gamma_I^x, \tag{2.34}$$

$$-\partial_x u_J + B\Big(u_J, \varphi_{J,1}^x, \ldots, \varphi_{J,N}^x\Big) = g_J^x, \quad \text{on } \Gamma_J^x, \tag{2.35}$$

where u_I and u_J are the main fields defined on Ω_I and Ω_J, respectively. The auxiliary fields $\{\varphi_{I,1}^x, \ldots, \varphi_{I,N}^x\} := \{\varphi_{IJ,1}, \ldots, \varphi_{IJ,N}\}$ and $\{\varphi_{J,1}^x, \ldots, \varphi_{J,N}^x\} := \{\varphi_{JI,1}, \ldots, \varphi_{JI,N}\}$ defined on the shared face are governed by a three-dimensional version of equation (2.19). We enforce weakly the continuity of transmission conditions at the shared face, which leads to

$$g_I^x = -g_J^x + 2B\Big(u_J, \varphi_{J,1}^x, \ldots, \varphi_{J,N}^x\Big), \quad \text{on } \Gamma_I^x, \tag{2.36}$$

$$g_J^x = -g_I^x + 2B\Big(u_I, \varphi_{I,1}^x, \ldots, \varphi_{I,N}^x\Big), \quad \text{on } \Gamma_J^x. \tag{2.37}$$

According to [70], because of the second-order spatial derivative in the governing equation on the shared face, the auxiliary fields require boundary conditions on the edges. At the edges $\Upsilon_I^{xy} = \Upsilon_J^{xy}$ and $\Upsilon_I^{xz} = \Upsilon_J^{xz}$, we use the transmission conditions

$$\partial_x \varphi_{I,j}^y + B\Big(\varphi_{I,j}^y, \psi_{I,1j}^{xy}, \ldots, \psi_{I,Nj}^{xy}\Big) = g_{I,j}^{xy}, \quad \text{on } \Upsilon_I^{xy}, \tag{2.38}$$

$$\partial_x \varphi_{I,k}^z + B\Big(\varphi_{I,k}^z, \psi_{I,1k}^{xz}, \ldots, \psi_{I,Nk}^{xz}\Big) = g_{I,k}^{xz}, \quad \text{on } \Upsilon_I^{xz}, \tag{2.39}$$

$$-\partial_x \varphi_{J,j}^y + B\Big(\varphi_{J,j}^y, \psi_{J,1j}^{xy}, \ldots, \psi_{J,Nj}^{xy}\Big) = g_{J,j}^{xy}, \quad \text{on } \Upsilon_J^{xy}, \tag{2.40}$$

$$-\partial_x \varphi_{J,k}^z + B\Big(\varphi_{J,k}^z, \psi_{J,1k}^{xz}, \ldots, \psi_{J,Nk}^{xz}\Big) = g_{J,k}^{xz}, \quad \text{on } \Upsilon_J^{xz}, \tag{2.41}$$

with N^2 auxiliary fields defined on each edge and governed by second-order PDEs. We enforce weakly the continuity of transmission conditions at the shared edges, which leads to

$$g_{J,j}^{xy} = -g_{I,j}^{xy} + 2B\Big(\varphi_{I,j}^y, \psi_{I,1j}^{xy}, \ldots, \psi_{I,Nj}^{xy}\Big), \quad \text{on } \Upsilon_J^{xy}, \tag{2.42}$$

$$g_{J,k}^{xz} = -g_{I,k}^{xz} + 2B\Big(\varphi_{I,k}^z, \psi_{I,1k}^{xz}, \ldots, \psi_{I,Nk}^{xz}\Big), \quad \text{on } \Upsilon_J^{xz}, \tag{2.43}$$

$$g_{I,j}^{xy} = -g_{J,j}^{xy} + 2B\Big(\varphi_{J,j}^y, \psi_{J,1j}^{xy}, \ldots, \psi_{J,Nj}^{xy}\Big), \quad \text{on } \Upsilon_I^{xy}, \tag{2.44}$$

$$g_{I,k}^{xz} = -g_{J,k}^{xz} + 2B\Big(\varphi_{J,k}^z, \psi_{J,1k}^{xz}, \ldots, \psi_{J,Nk}^{xz}\Big), \quad \text{on } \Upsilon_I^{xz}, \tag{2.45}$$

Again, because of these second-order PDEs, boundary conditions must be prescribed at the corner on the auxiliary fields belonging to the edges. We use the transmission conditions

$$\partial_x \varphi_{I,jk}^{yz} + B\Big(\varphi_{I,jk}^{yz}, \psi_{I,1jk}^{xyz}, \dots, \psi_{I,Njk}^{xyz}\Big) = g_{I,jk}^{xyz}, \quad \text{on } P_I^{xyz}, \tag{2.46}$$

$$-\partial_x \varphi_{J,jk}^{yz} + B\Big(\varphi_{J,jk}^{yz}, \psi_{J,1jk}^{xyz}, \dots, \psi_{I,Njk}^{xyz}\Big) = g_{J,jk}^{xyz}, \quad \text{on } P_J^{xyz}, \tag{2.47}$$

for $j,k = 1 \dots N$, with the scalar variables $\{\psi_{I,ijk}^{xyz}\}_{ijk}$ and $\{\psi_{J,ijk}^{xyz}\}_{ijk}$ defined as

$$\psi_{I,ijk}^{xyz} = -\frac{\alpha^2(c_i+1)\varphi_{I,jk}^{yz} + \alpha^2(c_j+1)\varphi_{I,ik}^{xz} + \alpha^2(c_k+1)\varphi_{I,ij}^{xy}}{\alpha^2 c_i + \alpha^2 c_j + \alpha^2 c_k + 1}, \quad \text{at } P_I^{xyz}, \tag{2.48}$$

$$\psi_{J,ijk}^{xyz} = -\frac{\alpha^2(c_i+1)\varphi_{J,jk}^{yz} + \alpha^2(c_j+1)\varphi_{J,ik}^{xz} + \alpha^2(c_k+1)\varphi_{J,ij}^{xy}}{\alpha^2 c_i + \alpha^2 c_j + \alpha^2 c_k + 1}, \quad \text{at } P_J^{xyz}, \tag{2.49}$$

for $i,j,k = 1 \dots N$. At the corner $P_I^{xyz} = P_J^{xyz}$, the new transmission variables $\{\varphi_{I,jk}^{yz}\}_{jk}$ and $\{\varphi_{J,jk}^{yz}\}_{jk}$ verify

$$g_{J,jk}^{xyz} = -g_{I,jk}^{xyz} + 2B\Big(\varphi_{I,jk}^{yz}, \psi_{I,1jk}^{xyz}, \dots, \psi_{I,Njk}^{xyz}\Big), \quad \text{on } P_J^{xyz}, \tag{2.50}$$

$$g_{I,jk}^{xyz} = -g_{J,jk}^{xyz} + 2B\Big(\varphi_{J,jk}^{yz}, \psi_{J,1jk}^{xyz}, \dots, \psi_{I,Njk}^{xyz}\Big), \quad \text{on } P_I^{xyz}, \tag{2.51}$$

for $j,k = 1 \dots N$. The iterative DD algorithm is similar to the algorithm in two dimensions. At each iteration, subproblems associated with the subdomains are solved parallelly, and transmission variables at the shared face, edges, and corners are updated. Here, the subproblem of Ω_I consists in finding the main field verifying the main system and auxiliary fields verifying the second-order PDEs on the shared face and edges. The transmission variables are associated to the shared face and edges, which are updated with formulas

$$g_I^{x\,(n+1)} = -g_J^{x\,(n)} + 2B\Big(u_J^{(n)}, \varphi_{J,1}^{x\,(n)}, \dots, \varphi_{J,N}^{x\,(n)}\Big), \quad \text{on } \Gamma_I^x, \tag{2.52}$$

and

$$g_{I,j}^{xy\,(n+1)} = -g_{J,j}^{xy\,(n)} + 2B\Big(\varphi_{J,j}^{y\,(n)}, \psi_{J,1j}^{xy\,(n)}, \dots, \psi_{J,Nj}^{xy\,(n)}\Big), \quad \text{on } \Upsilon_I^{xy}, \tag{2.53}$$

$$g_{I,k}^{xz\,(n+1)} = -g_{J,k}^{xz\,(n)} + 2B\Big(\varphi_{J,k}^{z\,(n)}, \psi_{J,1k}^{xz\,(n)}, \dots, \psi_{J,Nk}^{xz\,(n)}\Big), \quad \text{on } \Upsilon_I^{xz}, \tag{2.54}$$

and

$$g_{I,jk}^{xyz\,(n+1)} = -g_{J,jk}^{xyz\,(n)} + 2B\Big(\varphi_{J,jk}^{yz\,(n)}, \psi_{J,1jk}^{xyz\,(n)}, \dots, \psi_{J,Njk}^{xyz\,(n)}\Big), \quad \text{on } P_I^{xyz}, \tag{2.55}$$

For complete update formulas between Ω_I and Ω_J, we define g_{IJ} as

$$g_{IJ} := \left[\begin{array}{cccccccccc} g_I^x & g_{I,j}^{xy} & g_{I,k}^{xz} & g_{I,j}^{xy'} & g_{I,k}^{xz'} & g_{I,jk}^{xyz} & g_{I,jk}^{xyz'} & g_{I,jk}^{xy'z} & g_{I,jk}^{xy'z'} \end{array} \right]^\top, \tag{2.56}$$

where $g_{I,j}^{xy'}, g_{I,j}^{xz'}, g_{I,jk}^{xyz'}, g_{I,jk}^{xy'z}$, and $g_{I,jk}^{xy'z'}$ are transmission variables on $\Upsilon_I^{xy'}$, $\Upsilon_I^{xz'}$, and at $P^{xyz'}$, $P^{xy'z}$, $P^{xy'z'}$, respectively. Finally, considering all transmission variables associated with Ω_I and Ω_J, we have

$$g_{IJ}^{(n+1)} = -g_{JI}^{(n)} + 2 \left[\begin{array}{l} B\left(u_J^{(n)}, \varphi_{J,1}^{x\,(n)}, \ldots, \varphi_{J,N}^{x\,(n)}\right) \\ B\left(\varphi_{J,j}^{y\,(n)}, \psi_{J,1j}^{xy\,(n)}, \ldots, \psi_{J,Nj}^{xy\,(n)}\right) \\ B\left(\varphi_{J,k}^{z\,(n)}, \psi_{J,1k}^{xz\,(n)}, \ldots, \psi_{J,Nk}^{xz\,(n)}\right) \\ B\left(\varphi_{J,j}^{y'\,(n)}, \psi_{J,1j}^{xy'\,(n)}, \ldots, \psi_{J,Nj}^{xy'\,(n)}\right) \\ B\left(\varphi_{J,k}^{z'\,(n)}, \psi_{J,1k}^{xz'\,(n)}, \ldots, \psi_{J,Nk}^{xz'\,(n)}\right) \\ B\left(\varphi_{J,jk}^{yz\,(n)}, \psi_{J,1jk}^{xyz\,(n)}, \ldots, \psi_{J,Njk}^{xyz\,(n)}\right) \\ B\left(\varphi_{J,jk}^{yz'\,(n)}, \psi_{J,1jk}^{xyz'\,(n)}, \ldots, \psi_{J,Njk}^{xyz'\,(n)}\right) \\ B\left(\varphi_{J,jk}^{y'z\,(n)}, \psi_{J,1jk}^{xy'z\,(n)}, \ldots, \psi_{J,Njk}^{xy'z\,(n)}\right) \\ B\left(\varphi_{J,jk}^{y'z'\,(n)}, \psi_{J,1jk}^{xy'z'\,(n)}, \ldots, \psi_{J,Njk}^{xy'z'\,(n)}\right) \end{array} \right]. \tag{2.57}$$

Once again, this formula is similar to the general update formula (2.14), and all transmission variables $g_{IJ}^{(n+1)}$ ($I = 1 \ldots N_{\text{dom}}$, $J \in D_I$) can be merged into a global vector $\mathbf{g}^{(n+1)}$. The global process can be recast as one linear system, which can again be solved using Krylov subspace methods.

2.5.5 Convergence tests in 2D

In order to analyze the convergence rate of the DD method with compatibility relations in 2D, we first consider the same heterogeneous waveguide as in Section 2.4.2, with the only difference that we apply a Padé-type HABC on the right boundary so that the cross-point treatment is uniform. The number of auxiliary fields $N = 8$ and the parameter $\phi = \pi/3$ are used for both exterior and transmission conditions, and the frequency is $\omega = 20\pi$. The computational domain is partitioned into an grid of 16×1, 8×2 and 4×4 rectangular subdomains, with second order polynomial shape functions and 10 elements per wavelength.

The residual histories obtained with the three different decompositions are shown in Figure 2.20. The relative residual suddenly drops at the 15$^{\text{th}}$

(A) 1st iteration.

(B) 2nd iteration.

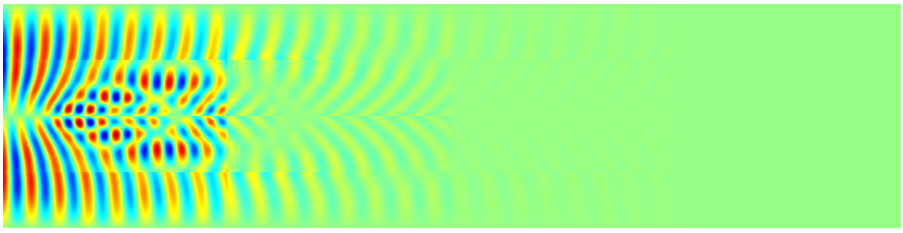
(C) 3rd iteration.

(D) 4th iteration.

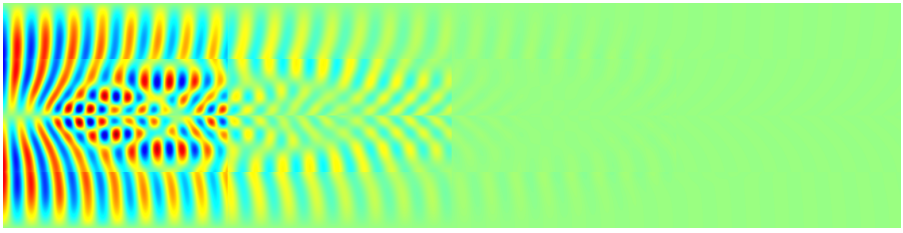
(E) 5th iteration.

FIGURE 2.18: Heterogeneous waveguide problem in two dimensions with 16×1 partitions. Snapshot of the solution after 1, 2, 3, 4 and 5 GMRES iterations.

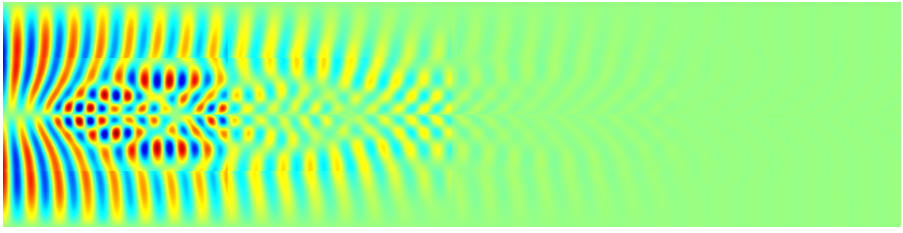
(A) 1st iteration.



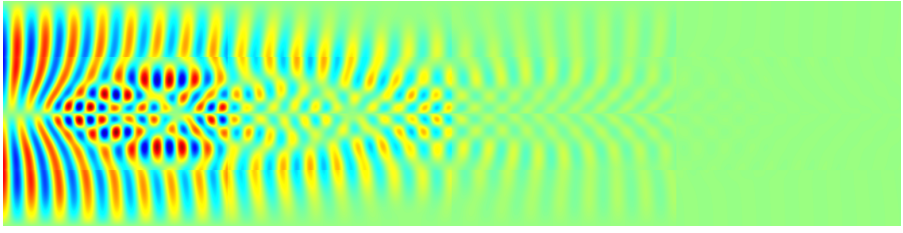
(B) 2nd iteration.



(C) 3rd iteration.



(D) 4th iteration.



(E) 5th iteration.

FIGURE 2.19: Heterogeneous waveguide problem in two dimensions with 4×4 partitions. Snapshot of the solution after 1, 2, 3, 4 and 5 GMRES iterations.

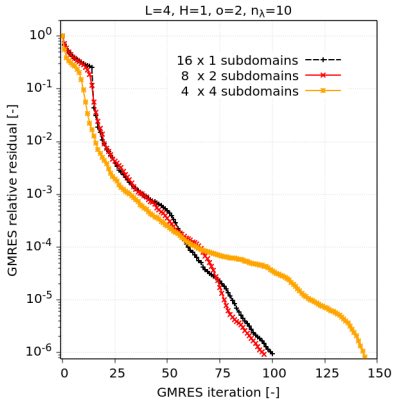


FIGURE 2.20: Heterogeneous waveguide problem. Residual history with second-order rectangular elements and 10 elements by wavelength.

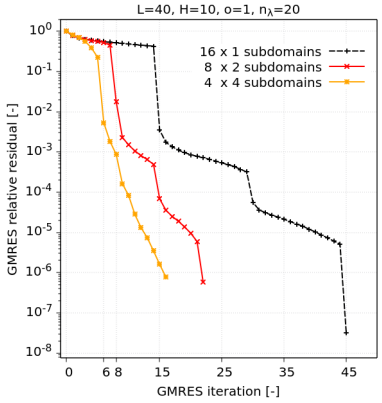
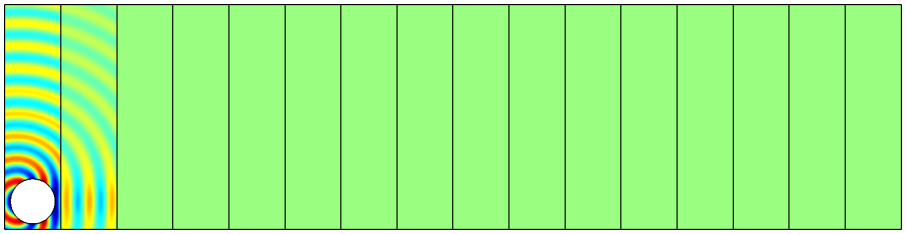
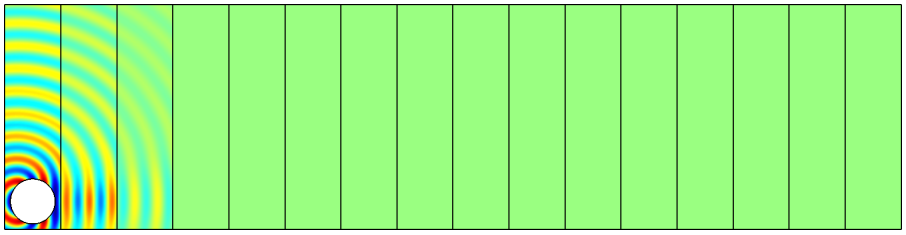


FIGURE 2.21: Scattering of a plane wave by a disk. Residual history with first order finite elements and 20 elements by wavelength.

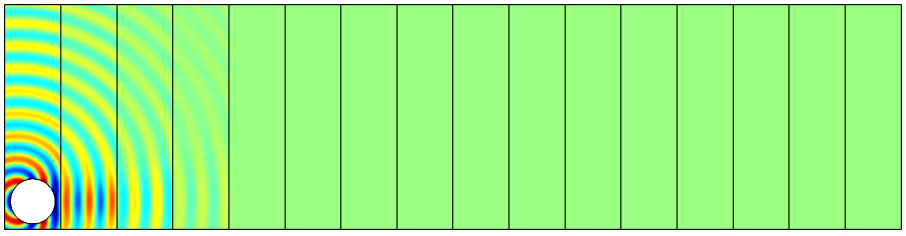
iteration with the 16×1 decomposition, when the wave reaches the final subdomain. With the other 8×2 and 4×4 decomposition, the residual decreases faster initially, while the convergence for the 4×4 eventually slows down faster than the others. Figures 2.18 and 2.19 help explain this behavior, by showing snapshots of the solutions after 1, 2, 3, 4 and 5 GMRES iterations with the 16×1 patition ant the 4×4 partition. For the 16×1 partition, one clearly see that the guided wave propagates in the horizontal direction, so that the one-way layered partition in this particular case follows the guided wave's properties. On the contray, the DD method in the 4×4 decomposition needs to resolve both horizontal and vertical reflections across interfaces, which is less suited for this guided wave problem and again illustrates the need for a



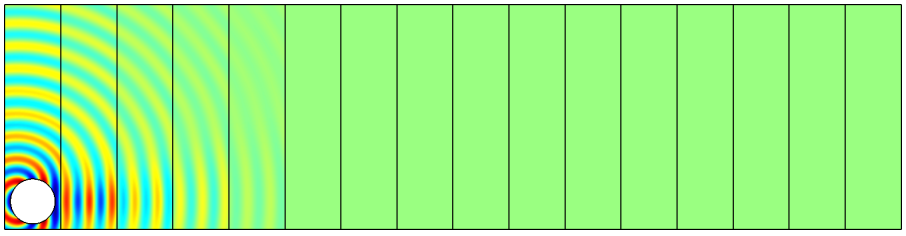
(A) 1st iteration.



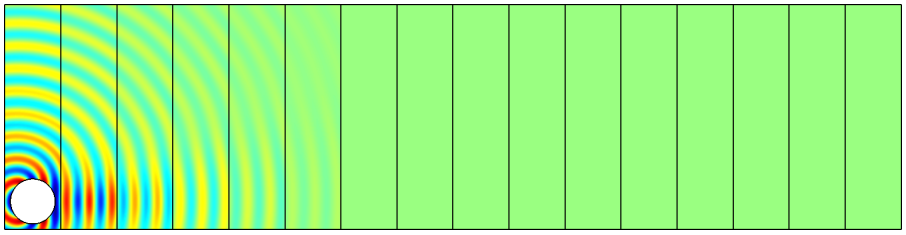
(B) 2nd iteration.



(C) 3rd iteration.

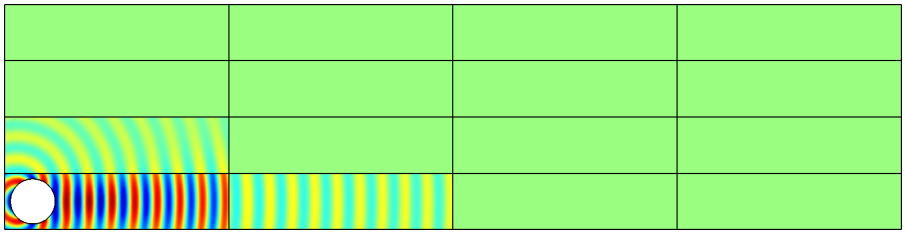


(D) 4th iteration.

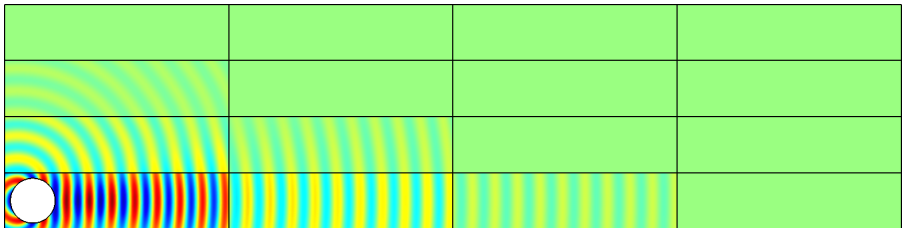


(E) 5th iteration.

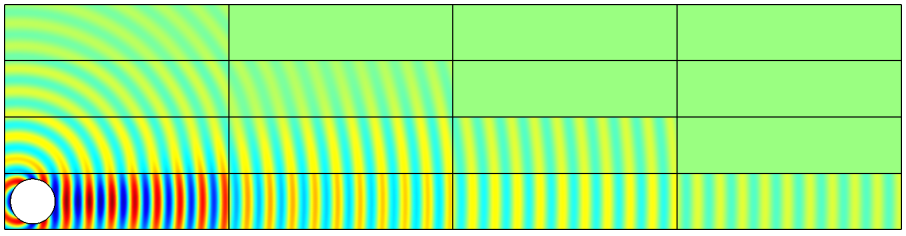
FIGURE 2.22: Scattering problem in two dimensions with 16×1 partitions. Snapshot of the solution after 1, 2, 3, 4 and 5 GMRES iterations.



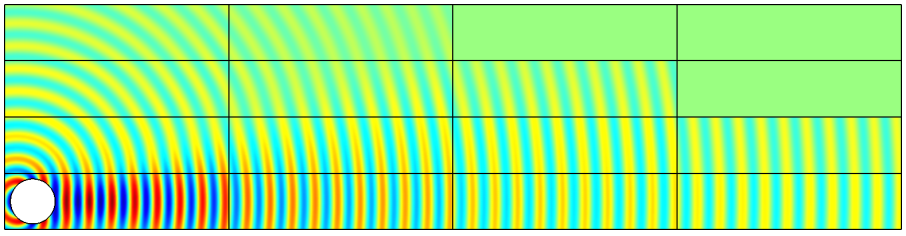
(A) 1st iteration.



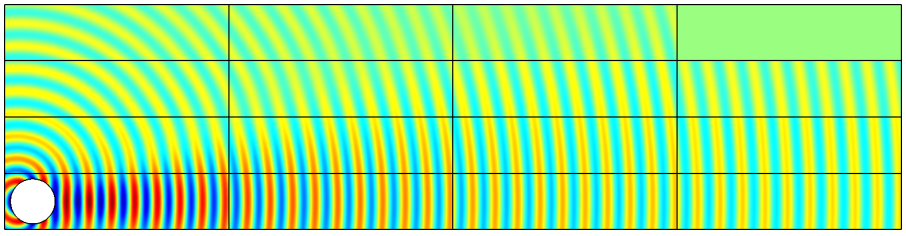
(B) 2nd iteration.



(C) 3rd iteration.



(D) 4th iteration.



(E) 5th iteration.

FIGURE 2.23: Scattering problem in two dimensions with 4×4 partitions. Snapshot of the solution after 1, 2, 3, 4 and 5 GMRES iterations.

robust preconditioning strategy.

The second problem that we consider is the scattering of an incident plane wave on a sound-soft disk. The geometry is a two-dimensional rectangular domain of size 40×10, which is partitioned into an grid of 16×1, 8×2 and 4×4 rectangular subdomains, with the disk placed on the bottom left. The Dirichlet boundary condition $u(\mathbf{x}) = -e^{ikx}$ is prescribed on the boundary of the disk, and a the Padé-type HABC is used on the exterior boundary and the artificial interfaces, with compatibility conditions at the corners. As in the previous test case, the number of auxiliary fields $N = 8$ and the parameter $\phi = \pi/3$ are used for both exterior and transmission conditions, and the frequency is $\omega = 20\pi$. First order finite elements are used, with 20 elements per wavelength.

The relative residual histories obtained with the three different decompositions are shown in Figure 2.21. The relative residual suddenly drops with all configurations at different iterations, when the wave reaches the final subdomain: at the 15^{th} iteration with the the 16×1 decomposition, at the 4×4 decomposition. This is consistent with the observations for the heterogeneous case: this scattering problem in free space doesn't involve any reflections, which lead to low iteration counts; moreover, there is no preferential direction for wave propagation, which makes the one-way layered decomposition less effective than the block decompositions.

2.5.6 Convergence tests in 3D

For 3D cases we only consider the problem of the scattering of a plane wave by a sound-soft sphere of radius equal to 1. We consider three different computational domains: a parallelepiped of size $2.5 \times 2.5 \times 20.0$, which is partitioned into $1 \times 1 \times 8$ subdomains; a parallelepiped of size $2.5 \times 5.0 \times 10.0$, which is partitioned into $1 \times 2 \times 4$ subdomains; and a parallelepiped of size $5.0 \times 5.0 \times 5.0$, which is partitioned into $2 \times 2 \times 2$ subdomains. The scattering sphere is placed at the center of the first subdomain (see Figure 2.24). The Dirichlet boundary condition $u(\mathbf{x}) = -e^{i\boldsymbol{\kappa}\cdot\mathbf{x}}$ is prescribed on the boundary of the sphere, and the Padé-type HABC is prescribed on the exterior boundary and the artificial interfaces, with compatibility conditions on the edges and at the corners. The number of auxiliary fields $N = 4$ and the parameter $\phi = \pi/3$ are used for both exterior and interior transmission conditions. The wavenumber $||\boldsymbol{\kappa}||$ is equal 2π with direction $[1/\sqrt{2}, 1/\sqrt{2}, 0]$. The solution is computed using on a tetrahedral mesh at order 7, with 3 elements per wavelength.

The same behavior as in the two-dimensional cases is observed, with sudden drops in the residual when the wave reaches the last subdomain, furthest from the scattering sphere (see Figure 2.25): at the 7^{th} iteration for the $1 \times 1 \times 8$ decomposition, at the 4^{th} iteration for the $1 \times 2 \times 4$ decomposition, and at the

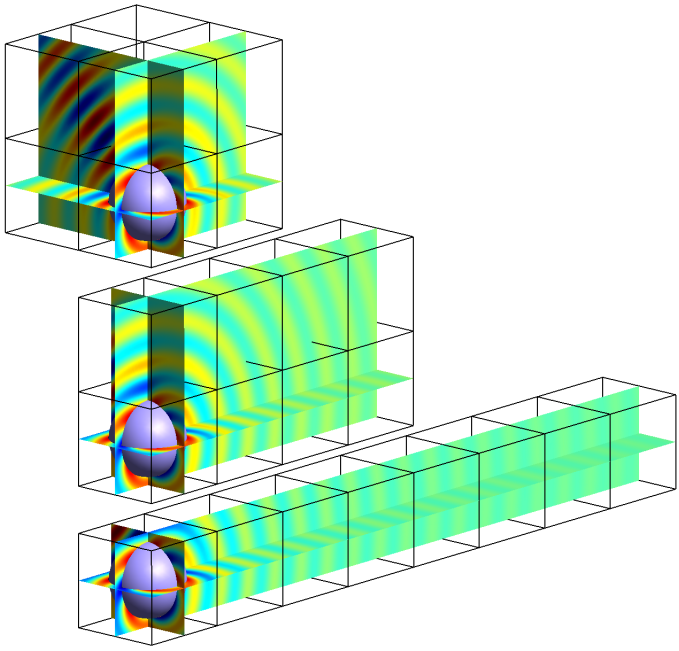


FIGURE 2.24: Scattering of a plane wave by a sphere. Snapshot of the solution for configurations with $1 \times 1 \times 8$, $1 \times 2 \times 4$ and $2 \times 2 \times 2$ subdomains (from bottom to top).

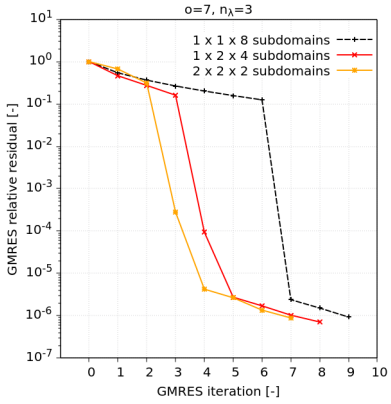
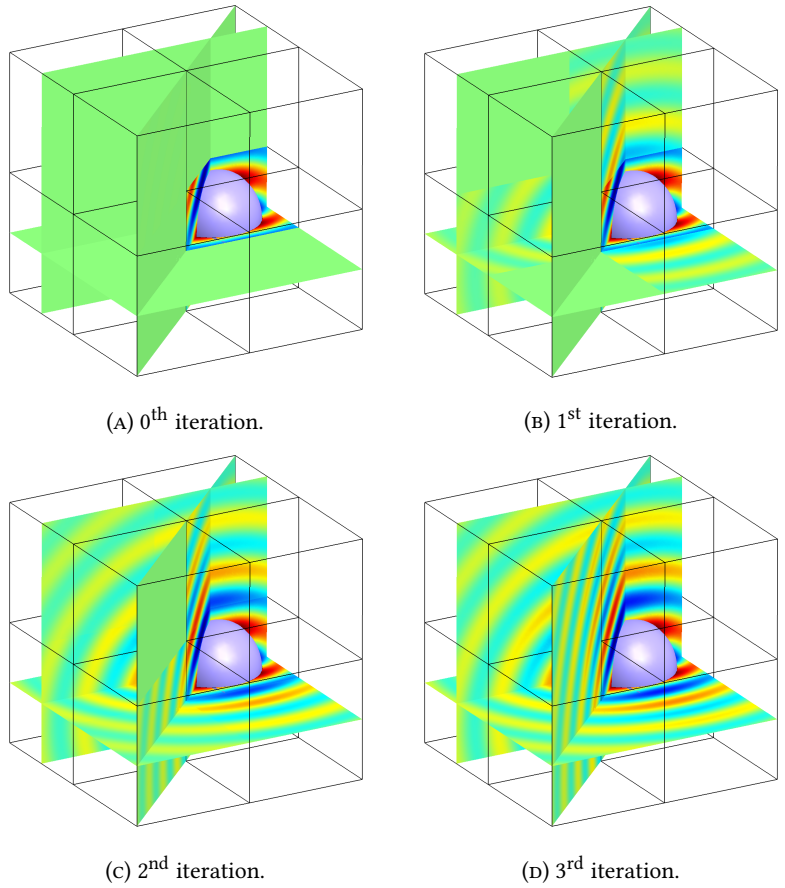


FIGURE 2.25: Scattering of a plane wave by a sphere. Residual history with order 7 tetrahedra and 3 elements by wavelength.

3^{rd} iteration for the $2 \times 2 \times 2$ decomposition. Again, the more directions the waves propagate in, the faster the rate of convergence (see Figure 2.26).



(A) 0^{th} iteration. (B) 1^{st} iteration.

(C) 2^{nd} iteration. (D) 3^{rd} iteration.

FIGURE 2.26: Scattering problem in three dimensions with $2 \times 2 \times 2$ partitions. Snapshot of the solution after 0, 1, 2, 3 GMRES iterations.

2.6 Conclusion

In this chapter we have analyzed the behavior of optimized Schwarz DD methods with both layered and block-type decompositions, the latter requiring special compatibility conditions at cross-points. Both decompositions can deal with large-scale problems in terms of memory usage and factorization times of the resulting discrete systems. While layered decompositions can be useful for Helmholtz problems with preferential wave propagation directions, we have shown that block-type decompositions are better suited for more general wave propagation problems, such as scattering problems.

However, even with accurate high-order Padé-type HABC transmission conditions, the number of iterations of the Krylov solver eventually depends on the number of subdomains, as in these one-level DD methods information

can only be transferred from one subdomain to its neighboring subdomains at each iteration. This motivates the development of multi-level preconditioning strategies, which will be introduced in the next chapter.

Sweeping preconditioners for optimized Schwarz methods

3

Even with optimal transmission conditions, as is expected for a one-level method, the number of iterations of the optimized Schwarz DD methods presented in the previous chapter grows as the number of subdomains increases. Since DD methods are iterative methods it is thus natural to try to design robust preconditioners. A solution for certain classes of problems is to add a component to the algorithm that is known in the DD methods' community as a *"coarse grid"* [36], in effect a second-level to enable longer-range information exchange than the local sharing (from one subdomain to its neighbors) of the one-level DD methods. Nevertheless, the design of robust coarse grids is very challenging for wave-type problems because of the highly oscillatory behavior of the solution, and several approaches are currently investigated in the community (see e.g. [24, 10] and references herein). In this chapter, we focus on an alternative approach, sweeping preconditioners, which have been proposed and studied for convection-diffusion problems in the '90s [73, 74]. Twenty years later, B. Engquist and L. Ying [31] showed that the sweeping preconditioner they proposed could yield algorithms that are quasi-independent of the number of subdomains. Their work led to a class of sweeping preconditioners for high-frequency Helmholtz problems [84, 91, 96, 41, 87, 15], and since then sweeping preconditioners have garnered a lot of interest for high-frequency Helmholtz problems. However, sweeping preconditioners have two major drawbacks: they rely on intrinsically sequential operations (they are related to a LU-type factorization of the underlying iteration operator) and they are naturally only suited for layered-type domain decompositions (where the layered structure allows to explicit the LU factorization as a double sweep across the subdomains).

In this chapter, we explore a family of generalized sweeping preconditionners where sweeps can be done in several directions for non-overlapping block domain partitions (checkerboard-type in 2D, Rubick's cube-type in 3D). This contribution relies on the availability of transmission conditions able to deal with cross-points, such as the high-order Padé-type transmission conditions with and cross-point treatment presented in the previous chapter. We

will derive sweeping preconditioners in a systematic manner, based on the explicit representation of the iteration matrix in the case of such block decompositions. The sweeps can be performed in Cartesian and diagonal directions, and several sweeping directions can be combined by using the flexible version of GMRES [78, 79], which allows to change the preconditioner at each iteration. For applicative cases, the resulting preconditioners provide an effective way to rapidly transfer information in the different zones of the computational domain, accelerating the convergence of iterative solution procedure with GMRES. Our approach is related to the recent work on L-sweeps preconditioners [87] and diagonal sweeping technique [63], where directional sweeping strategies are proposed in the context of the method of polarized traces and the source transfer method, respectively. Here, the preconditioners are proposed for non-overlapping domain decomposition solvers with high-order transmission conditions, and several directions can be combined thanks to the use of flexible GMRES.

3.1 Algebraic structure of the interface problem

Let us start by analyzing the algebraic structure of the global interface problem (2.9), i.e.

$$\mathcal{F}\mathbf{g} := (\mathcal{I} - \mathcal{A})\mathbf{g} = \mathbf{b}, \tag{3.1}$$

where $\mathcal{A}$ is the iteration matrix, $\mathbf{g}$ is the set of all transmission variables and $\mathbf{b}$ is given by the source term. The global matrix $\mathcal{F} := \mathcal{I} - \mathcal{A}$ can be represented as a $N_{\text{dom}} \times N_{\text{dom}}$ sparse block matrix, which each block corresponds to the coupling between the unknowns of two subdomains. The nature of the blocks and the sparse structure of the global block matrix are analyzed in the next two sections.

3.1.1 Identification of the blocks

Using the block representation, the abstract system (3.1) can be rewritten as

$$\sum_{J=1}^{N_{\text{dom}}} \mathcal{F}_I^J \mathbf{g}_J = \mathbf{b}_I, \quad I = 1 \dots N_{\text{dom}}, \tag{3.2}$$

where the vectors $\mathbf{g}_I$ and $\mathbf{b}_I$ contain all the transmission variables and the source terms, respectively, for the subdomain Ω_I. The block $\mathcal{F}_I^J$ corresponds to a coupling between the transmission variables of the subdomains Ω_I and Ω_J. The blocks corresponding to subdomains that are not neighbours (*i.e* which do not share any interface edge) are cancelled because there is no direct coupling between the corresponding variables. Since there are at most four

neighbouring subdomains for each subdomains, there are at most four off-diagonal blocks in each line and each column of the global block matrix.

For studying the blocks, we consider a setting with transmission conditions based on the basic impedance operator for the sake of simplicity. In that case, all the transmission variables are associated to the interface edges. Since there are two transmission variables per interface edge (one for each neighboring subdomain), the total size of the vectors $\mathbf{g}$ and $\mathbf{b}$ is twice the number of interface edges. The size of the blocks $\mathbf{g}_I$ and $\mathbf{b}_I$ in these vectors corresponds to the number of interface edges for the subdomain Ω_I. The block can then be rectangular if the neighbouring subdomains Ω_I and Ω_J have different numbers of interface edges. To simplify the presentation, we assume hereafter that Ω_I and Ω_J do not touch the exterior border of the main domain (i.e. each of them has four interface edges and $\mathcal{F}_I^J$ is a 4×4 matrix).

Every line of the system corresponds to a relation similar to equation (2.13). Let us consider the line for the transmission variable g_{IJ}, which corresponds to the relation

$$g_{IJ} = -g_{JI} + 2\,\mathcal{B}_{JI} u_J, \quad \text{on } \Gamma_{IJ}, \tag{3.3}$$

where J is such that Ω_I and Ω_J are neighbouring subdomains with the shared interface edge $\Gamma_{IJ} = \Gamma_{JI}$. By the linearity of the problem, the solution u_J can be slip into two contributions, $u_J = v_J + w_J$. The field v_J is the solution of subproblem (2.11) for Ω_J where the right-hand-side term of the Dirichlet boundary condition on $\partial\Omega_J \cap \Gamma_{\text{sca}}$ is cancelled. The field w_J is the solution of subproblem (2.11) for Ω_J where the right-hand-side terms of the transmission conditions are cancelled. Equation (3.3) can then be rewritten as

$$g_{IJ} = -g_{JI} + 2\,\mathcal{B}_{JI} v_J + b_{JI}, \quad \text{on } \Gamma_{IJ}, \tag{3.4}$$

where $b_{JI} := 2\mathcal{B}_{JI} w_J$ depends only on the data of the problem, with is the incident plane wave is the present case. In order to exhibit dependences between transmission variables, we decompose the field v_J into several contributions. For every interface edge Γ_{JK} (with $K \in D_J$), we introduce the field $v_{JK}(g_{JK})$ as the solution of subproblem (2.11) for Ω_J with the transmission variable g_{JK} prescribed on Γ_{JK} and where all the other transmission variables and the right-hand-side term of the Dirichlet condition are cancelled. By linearity, the solution of v_J can then be written as

$$v_J = \sum_{K \in D_J} v_{JK}(g_{JK}), \tag{3.5}$$

Using this decomposition into equation (3.4) gives

$$g_{IJ} + g_{JI} - 2 \sum_{K \in D_J} \mathcal{B}_{JI} v_{JK}(g_{JK}) = b_{JI}, \quad \text{on } \Gamma_{IJ}. \tag{3.6}$$

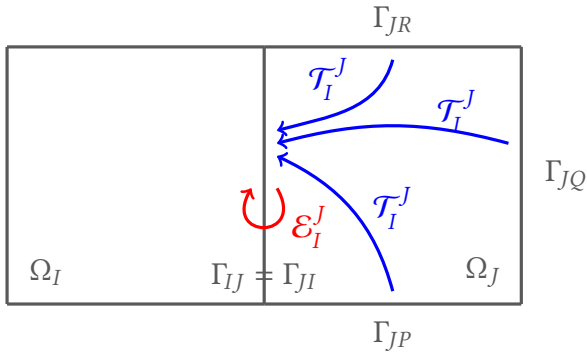


Figure 3.1: Illustration of the coupling introduced by the self-coupling operator (in red) and the transfer operator (in blue) for a configuration with neighboring subdomains Ω_I and Ω_J with the shared interface edge $\Gamma_{IJ} = \Gamma_{JI}$.

Defining the self-coupling and transfer operators as

$$\begin{aligned} \mathcal{E}_I^J : \quad g_{JI} &\longmapsto \quad g_{JI} - 2\mathcal{B}_{JI} v_{JI}(g_{JI}) \;:=\; \mathcal{E}_I^J g_{JI}, \\ \mathcal{T}_I^J : \quad g_{JK} &\longmapsto \quad -2\mathcal{B}_{JI} v_{JK}(g_{JK}) \;:=\; \mathcal{T}_I^J g_{JK}, \quad \text{for } K \in D_J, K \neq I, \end{aligned} \tag{3.7}$$

we finally have the representation

$$g_{IJ} + \mathcal{E}_I^J g_{JI} + \sum_{K \neq I} \mathcal{T}_I^J g_{JK} = b_{JI}, \quad \text{on } \Gamma_{IJ}. \tag{3.8}$$

The self-coupling operator $\mathcal{E}_I^J$ introduces a coupling between the transmission variables living on the same interface edge $\Gamma_{IJ} = \Gamma_{JI}$, while the transfer operators introduces a coupling between g_{IJ} and the transmission variables living on the other interface edges of Ω_J (i.e. any $\Gamma_{JK} \neq \Gamma_{IJ}$). These couplings are illustrated in Figure 3.1.

Thanks to the representation in equation (3.8), the elements of the matrix and the right-hand side of the global system can be identified. The right-hand side of (3.8) is an element of $\mathbf{b}_I$. Looking at the first term in the left-hand side, we straightforwardly have that the blocks on the diagonal of the global matrix are identity matrices. Finally, the second and third terms correspond to elements in the off-diagonal block $\mathcal{F}_I^J$. The other elements of this block are equal to zero because the relation (3.8) corresponding to the other edges of Ω_I do not involve transmission variables of Ω_J. For instance, there are the neighbouring subdomains Ω_I and Ω_J, and four neighbouring subdomains of Ω_J are Ω_I, Ω_P, Ω_Q, Ω_R (see Figure 3.1). Assuming that the shared interface

edge is $\Gamma_{IJ} = \Gamma_{JI}$, the block $\mathcal{F}_I^J$ and the vector $\mathbf{g}_J$ read

$$\mathcal{F}_I^J = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \mathcal{E}_I^J & \mathcal{T}_I^J & \mathcal{T}_I^J & \mathcal{T}_I^J \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{and} \quad \mathbf{g}_J = \begin{bmatrix} g_{JI} \\ g_{JP} \\ g_{JQ} \\ g_{JR} \end{bmatrix}. \tag{3.9}$$

Using equation (3.8), one has the relation

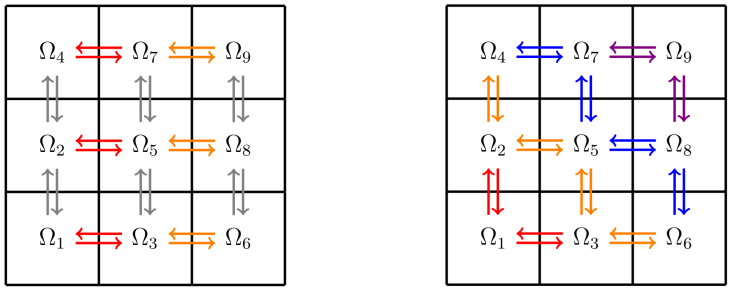
$$\begin{bmatrix} 0, & 0, & g_{IJ}, & 0 \end{bmatrix}^\top + \mathcal{F}_I^J \, \mathbf{g}_J = \begin{bmatrix} 0, & 0, & b_{JI}, & 0 \end{bmatrix}^\top. \tag{3.10}$$

This example corresponds to the illustration in Figure 3.1. In the general case with subdomains touching the exterior border of the main domain, this matrix can be rectangular with numbers of lines and columns between two and four. Each block always exhibits only one non-zero line, with one self-coupling operator and between one and three transfer operators.

3.1.2 Block matrix forms for the global system

The sparse structure of the global block matrix consists of all blocks $\mathcal{F}_I^J$ and identity blocks $\mathbf{I}_I$. With one-dimensional domain partitions, the matrix is block tridiagonal, which was leveraged to device efficient sweeping preconditioners (see e.g. [84, 91, 4]). With checkerboard partitions, the matrix can also be block tridiagonal if the blocks are arranged correctly. In order to clearly present this tridiagonal structure which will illustrate horizontal sweeps and diagonal sweeps in the next section, the subdomains are arranged in columns and diagonals, which are illustrated in Figure 3.2 for a 3×3 checkerboard partition.

We first analyze the structure of the global system obtained with the column-type arrangement. For the 3×3 checkerboard partition, the system



(a) Column-type arrangement (b) Diagonal-type arrangement

FIGURE 3.2: Illustration of the subdomains arranged in two different manners with a 3×3 checkerboard partition. The colored arrows indicate interactions between groups of subdomains.

can be written as

$$\left[\begin{array}{ccc|ccc|ccc} \mathbf{I}_1 & \mathcal{F}_1^2 & & \mathcal{F}_1^3 & & & & & \\ \mathcal{F}_2^1 & \mathbf{I}_2 & \mathcal{F}_2^4 & & \mathcal{F}_2^5 & & & & \\ & \mathcal{F}_4^2 & \mathbf{I}_4 & & & \mathcal{F}_4^7 & & & \\ \hline \mathcal{F}_3^1 & & & \mathbf{I}_3 & \mathcal{F}_3^5 & & \mathcal{F}_3^6 & & \\ & \mathcal{F}_5^2 & & \mathcal{F}_5^3 & \mathbf{I}_5 & \mathcal{F}_5^7 & & \mathcal{F}_5^8 & \\ & & \mathcal{F}_7^4 & & \mathcal{F}_7^5 & \mathbf{I}_7 & & & \mathcal{F}_7^9 \\ \hline & & & \mathcal{F}_6^3 & & & \mathbf{I}_6 & \mathcal{F}_6^8 & \\ & & & & \mathcal{F}_8^5 & & \mathcal{F}_8^6 & \mathbf{I}_8 & \mathcal{F}_8^9 \\ & & & & & \mathcal{F}_9^7 & & \mathcal{F}_9^8 & \mathbf{I}_9 \end{array}\right] \left[\begin{array}{c} \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_4 \\ \hline \mathbf{g}_3 \\ \mathbf{g}_5 \\ \mathbf{g}_7 \\ \hline \mathbf{g}_6 \\ \mathbf{g}_8 \\ \mathbf{g}_9 \end{array}\right] = \left[\begin{array}{c} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_4 \\ \hline \mathbf{b}_3 \\ \mathbf{b}_5 \\ \mathbf{b}_7 \\ \hline \mathbf{b}_6 \\ \mathbf{b}_8 \\ \mathbf{b}_9 \end{array}\right], \tag{3.11}$$

where $\mathbf{I}_I$ is the identity matrix associated to the subdomain Ω_I. The global matrix can be rewritten as a 3×3 block tridiagonal matrix, where each large block corresponds to interactions between those subdomains belonging to two given columns of the domain partition. Each large block contains 3×3 small blocks corresponding to interactions between the subdomains of both columns. The limits of these large blocks are represented with black lines in (3.11). The off-diagonal large blocks, which correspond to interactions between two different columns, are block diagonal. For a general $N_r \times N_c$ partition of the domain, the structure remains the same. The global matrix remains a block tridiagonal matrix with $N_c \times N_c$ large blocks, each large block

contains $N_r \times N_r$ small blocks, and the off-diagonal large blocks remains block diagonal.

With the diagonal-type arrangement, the structure of the global system can be written as

$$\begin{bmatrix}
\mathbf{I}_1 & \mathcal{F}_1^2 & \mathcal{F}_1^3 & & & & & & \\
\mathcal{F}_2^1 & \mathbf{I}_2 & & \mathcal{F}_2^4 & \mathcal{F}_2^5 & & & & \\
\mathcal{F}_3^1 & & \mathbf{I}_3 & & \mathcal{F}_3^5 & \mathcal{F}_3^6 & & & \\
 & \mathcal{F}_4^2 & & \mathbf{I}_4 & & & \mathcal{F}_4^7 & & \\
 & \mathcal{F}_5^2 & \mathcal{F}_5^3 & & \mathbf{I}_5 & & \mathcal{F}_5^7 & \mathcal{F}_5^8 & \\
 & & \mathcal{F}_6^3 & & & \mathbf{I}_6 & & \mathcal{F}_6^8 & \\
 & & & \mathcal{F}_7^4 & \mathcal{F}_7^5 & & \mathbf{I}_7 & & \mathcal{F}_7^9 \\
 & & & & \mathcal{F}_8^5 & \mathcal{F}_8^6 & & \mathbf{I}_8 & \mathcal{F}_8^9 \\
 & & & & & & \mathcal{F}_9^7 & \mathcal{F}_9^8 & \mathbf{I}_9
\end{bmatrix}
\begin{bmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \\ \mathbf{g}_4 \\ \mathbf{g}_5 \\ \mathbf{g}_6 \\ \mathbf{g}_7 \\ \mathbf{g}_8 \\ \mathbf{g}_9 \end{bmatrix}
=
\begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \\ \mathbf{b}_4 \\ \mathbf{b}_5 \\ \mathbf{b}_6 \\ \mathbf{b}_7 \\ \mathbf{b}_8 \\ \mathbf{b}_9 \end{bmatrix}. \tag{3.12}$$

The matrix of the system can again be written as a block tridiagonal matrix with large blocks corresponding to interactions between two groups of subdomains. Here, each group corresponds to the subdomains on a given diagonal of the domain partition (see Figure 3.2b). In the matrix, the diagonal large blocks are identity matrices because subdomains belonging to the same group are never neighbours. The off-diagonal large block are rectangular with different sizes because the groups contain different numbers of subdomains. For a general $N_r \times N_c$ partition of the domain, there are $N_r + N_c - 1$ groups of subdomains and the matrix of the system can be still be written as a block tridiagonal matrix.

Whatever the subdomains are grouped by column, row or diagonal, the global system can be represented with a bloc tridiagonal matrix. For convenience, we introduce the general representation

$$\begin{bmatrix}
\mathcal{F}_{[1]}^{[1]} & \mathcal{F}_{[1]}^{[2]} & & & \\
\mathcal{F}_{[2]}^{[1]} & \mathcal{F}_{[2]}^{[2]} & \mathcal{F}_{[2]}^{[3]} & & \\
 & \mathcal{F}_{[3]}^{[2]} & \mathcal{F}_{[3]}^{[3]} & \mathcal{F}_{[3]}^{[4]} & \\
 & & \mathcal{F}_{[4]}^{[3]} & \mathcal{F}_{[4]}^{[4]} & \mathcal{F}_{[4]}^{[5]} \\
 & & & \mathcal{F}_{[5]}^{[4]} & \mathcal{F}_{[5]}^{[5]}
\end{bmatrix}
\begin{bmatrix} \mathbf{g}_{[1]} \\ \mathbf{g}_{[2]} \\ \mathbf{g}_{[3]} \\ \mathbf{g}_{[4]} \\ \mathbf{g}_{[5]} \end{bmatrix}
=
\begin{bmatrix} \mathbf{b}_{[1]} \\ \mathbf{b}_{[2]} \\ \mathbf{b}_{[3]} \\ \mathbf{b}_{[4]} \\ \mathbf{b}_{[5]} \end{bmatrix}, \tag{3.13}$$

where the vectors $\mathbf{g}_{[I]}$ and $\mathbf{b}_{[I]}$ are associated to one group of subdomains and each block $\mathcal{F}_{[I]}^{[J]}$ corresponds to the coupling between the transmission

variables of two groups. Each block corresponds to one box in equations (3.11) and (3.12).

3.2 Sweeping preconditioners for the interface problem

In this section, we present sweeping preconditioners to accelerate the solution of the interface problem $\mathcal{F}\mathbf{g} = \mathbf{b}$ with standard iterative schemes based on Krylov subspaces. To be efficient, a preconditioner $\widetilde{\mathcal{F}}$ must be designed such that solving the preconditioned problem $\widetilde{\mathcal{F}}^{-1}\mathcal{F}\mathbf{g} = \widetilde{\mathcal{F}}^{-1}\mathbf{b}$ is faster than solving the unpreconditioned problem, and applying the inverse of the preconditioner on any vector is affordable. With sweeping preconditioners, applying the inverse of $\widetilde{\mathcal{F}}$ on a vector corresponds to solving subproblems in a certain order to transfer information following the natural path taken by propagative waves.

Sweeping preconditioners have been proposed for layered partitions of the domain e.g. in [75, 84, 91, 85, 92]. With this kind of partition, the global matrix can be written with a block tridiagonal representation, which each block on the diagonal is an identity matrix and each off-diagonal block is associated to the coupling between two neighboring layers. Thanks to this structure, the lower and upper triangular parts of the global matrix, which are used in standard Gauss-Seidel and SOR preconditioners, can be explicitly inverted. Applying the inverse of the lower and upper triangular matrices simply corresponds to solving subproblems following forward and backward sweeps over the subdomains, respectively. This approach has been used in [92] to design various sweeping preconditioners for layered partitions.

We propose an extension of sweeping preconditioners for checkerboard partitions. Ideas used in [92] are applied here by considering the block tridiagonal representation of the global matrix shown in equation (3.13). Although the sweeps are performed over the groups of subdomains, the sweeping directions don't depend on the arrangement of the subdomains. Column-type and diagonal-type arrangement of the subdomains are only used to illustrate horizontal and diagonal sweeps. For the sake of clarity, we introduce the groups of subdomains $\Omega_{[S]}$ ($S = 1 \dots N_{\text{gr}}$), which correspond to columns or diagonals in Figure 3.2. Block symmetric Gauss-Seidel (SGS) and parallel double sweep (DS) preconditioners are described in section 3.2.1 and 3.2.2. Computational aspects and extensions are discussed in section 3.2.3.

3.2.1 Block Symmetric Gauss-Seidel (SGS) preconditioner

The general block Symmetric Gauss-Seidel (SGS) preconditioner reads

$$\widetilde{\mathcal{F}}_{\text{SGS}} = (\mathcal{D} + \mathring{\mathcal{L}})\mathcal{D}^{-1}(\mathcal{D} + \mathring{\mathcal{U}}), \tag{3.14}$$

where $\mathcal{D}$, $\mathring{\mathcal{L}}$ and $\mathring{\mathcal{U}}$ are respectively the diagonal part, the strictly lower triangular part and the strictly upper triangular part of the block matrix $\mathcal{F}$ represented in equation (3.13). Assuming there is no coupling between subdomains of the same group, the diagonal part is an identity matrix, $\mathcal{D} = \mathcal{I}$. The preconditioner can then be rewritten as $\widetilde{\mathcal{F}}_{\text{SGS}} = \mathcal{L}\mathcal{U}$, with the lower triangular matrix $\mathcal{L} = \mathcal{I} + \mathring{\mathcal{L}}$ and the upper triangular matrix $\mathcal{U} = \mathcal{I} + \mathring{\mathcal{U}}$, which reads

$$\mathcal{L} = \begin{bmatrix} \mathcal{I} & & & & \\ \mathcal{F}^{[1]}_{[2]} & \mathcal{I} & & & \\ & \mathcal{F}^{[2]}_{[3]} & \mathcal{I} & & \\ & & \ddots & \ddots & \\ & & & \mathcal{F}^{[N_{\text{gr}}-1]}_{[N_{\text{gr}}]} & \mathcal{I} \end{bmatrix}, \mathcal{U} = \begin{bmatrix} \mathcal{I} & \mathcal{F}^{[2]}_{[1]} & & & \\ & \mathcal{I} & \mathcal{F}^{[3]}_{[2]} & & \\ & & \ddots & \ddots & \\ & & & \mathcal{I} & \mathcal{F}^{[N_{\text{gr}}]}_{[N_{\text{gr}}-1]} \\ & & & & \mathcal{I} \end{bmatrix}.$$

To compute explicitly the inverse of the preconditioner, we introduce the matrices $\mathcal{L}_{([I],[J])}$ and $\mathcal{U}_{([I],[J])}$ that contain the unit diagonal and only one off-diagonal block with the element $\mathcal{F}^{[J]}_{[I]}$. We have

$$\begin{aligned} \widetilde{\mathcal{F}}_{\text{SGS}} =& \mathcal{L}\,\mathcal{U} \\ =& \Big(\mathcal{L}_{([2],[1])}\,\mathcal{L}_{([3],[2])}\,\cdots\,\mathcal{L}_{([N_{\text{gr}}],[N_{\text{gr}}-1])}\Big) \\ & \Big(\mathcal{U}_{([N_{\text{gr}}-1],[N_{\text{gr}}])}\,\cdots\,\mathcal{U}_{([2],[3])}\,\mathcal{U}_{([1],[2])}\Big) \end{aligned} \tag{3.15}$$

and

$$\begin{aligned} \widetilde{\mathcal{F}}^{-1}_{\text{SGS}} =& \mathcal{U}^{-1}\,\mathcal{L}^{-1} \\ =& \Big(\mathcal{U}^{-1}_{([1],[2])}\,\mathcal{U}^{-1}_{([2],[3])}\,\cdots\,\mathcal{U}^{-1}_{([N_{\text{gr}}-1],[N_{\text{gr}}])}\Big) \\ & \Big(\mathcal{L}^{-1}_{([N_{\text{gr}}],[N_{\text{gr}}-1])}\,\cdots\,\mathcal{L}^{-1}_{([3],[2])}\,\mathcal{L}^{-1}_{([2],[1])}\Big). \end{aligned} \tag{3.16}$$

We can easily see that the inverse of the matrices $\mathcal{L}_{([I],[J])}$ and $\mathcal{U}_{([I],[J])}$ are the same matrices, but with the opposite sign on the off-diagonal block. Applying the inverse of the preconditioner can be then computed using Algorithm 4. The procedure is written more explicitly in Algorithm 5.

In Algorithm 4, the application of $\mathcal{F}^{[S]}_{[S+1]}$ on a vector $\mathbf{r}_S$ corresponds to solving subproblems defined on the subdomains of $\Omega_{[S]}$ with the transmission data contained in $\mathbf{r}_S$. The result is used to update transmission variables in $\mathbf{r}_{S+1}$, associated to the subdomains of $\Omega_{[S+1]}$. Therefore, the first loop, which corresponds to the application of $\mathcal{L}^{-1}$, can be interpreted as a forward sweep, where information are propagated across groups of increasing number. Similarily, the loop corresponding to the application of $\mathcal{U}^{-1}$ can be interpreted as a backward sweep, where information are propagated across groups of decreasing number. The sweeps are performed in the horizontal or diagonal

Algorithm 4: Application of the SGS preconditioner: $\mathbf{r} \leftarrow \widetilde{\mathcal{F}}_{\text{SGS}}^{-1}\,\mathbf{r}$.

// Forward sweep (application of $\mathcal{L}^{-1}$)
for $S = 1 : (N_{gr} - 1)$ **do**
  $\mathbf{r}_{S+1} \leftarrow \mathbf{r}_{S+1} - \mathcal{F}_{[S+1]}^{[S]}\mathbf{r}_S$
end
// Backward sweep (application of $\mathcal{U}^{-1}$)
for $S = N_{gr} : 2$ **do**
  $\mathbf{r}_{S-1} \leftarrow \mathbf{r}_{S-1} - \mathcal{F}_{[S-1]}^{[S]}\mathbf{r}_S$
end

direction not depending on the arrangement of the subdomains. Iterations of the backward sweep are illustrated on Figures 3.3a and 3.3b for horizontal sweeps and diagonal sweeps.

In the SGS preconditioner, the forward and backward sweeps must be performed sequentially. Nevertheless, in each iteration of both loops, the subproblems of a given group of subdomains can be solved in parallel. With the horizontal sweeps, the update of the transmission variables inside each group have been avoided since the diagonal part $\mathcal{D}$ has been replaced with an identity matrix. With the diagonal sweeps, there is no coupling between the subdomains of the same group since there is no shared edge between them.

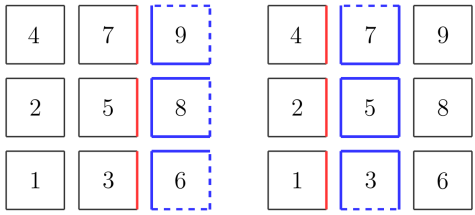
3.2.2 Parallel Double Sweep (DS) preconditioner

With the parallel Double Sweep (DS) preconditioner, the $\mathcal{L}$ and $\mathcal{U}$ matrices are modified in such a way that applying one of them does not modify the elements of the vector used by the other. The forward and backward sweeps can then be performed in parallel, without data race, which potentially reduces the runtime per iteration by a factor two in parallel environments.

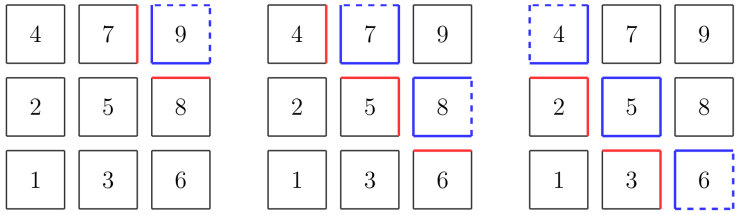
The DS preconditioner can be written as $\widetilde{\mathcal{F}}_{\text{DS}} = \widetilde{\mathcal{L}}\,\widetilde{\mathcal{U}} = \widetilde{\mathcal{U}}\,\widetilde{\mathcal{L}} = \widetilde{\mathcal{L}}+\widetilde{\mathcal{U}}-\mathcal{I}$. To remove the dependences between $\mathcal{L}$ and $\mathcal{U}$, the blocks are modified in such a way that, for a given group of subdomains $\Omega_{[S]}$, the forward sweep does not use transmission data from edges shared with a subdomain of $\Omega_{[S+1]}$ (these data are modified in the backward sweep), and the backward sweep does not use transmission data from edges shared with a subdomain of $\Omega_{[S-1]}$ (these data are modified in the forward sweep). The effective update process is illustrated in Figures 3.3c and 3.3d for both horizontal and diagonal sweeps. The application of the DS preconditioner on a vector is detailed in Algorithm 5. The main difference with the SGS preconditioner is that one or more trans-

Algorithm 5: Application of the SGS or DS preconditioner: $\mathbf{r} \leftarrow \widetilde{\mathcal{F}}^{-1}\mathbf{r}$.

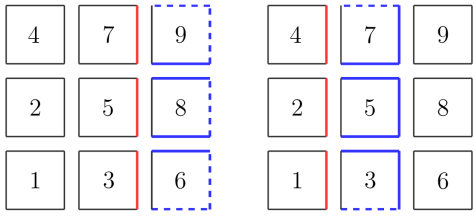
// Forward sweep
for $S = 1 : (N_{gr} - 1)$ **do**
  parfor *each* I *such that* $\Omega_I \subset \Omega_{[S]}$ **do**
    Configure boundary data for subproblem on Ω_I:
      $u_D \leftarrow 0$ on $\partial\Omega_I \cap \Gamma_D$
      $g_{IK} \leftarrow r_{IK}$ on each interface edge $\Gamma_{IK} \not\subset \partial\Omega_{[S+1]}$
      $g_{IK} \leftarrow r_{IK}$ on each interface edge $\Gamma_{IK} \subset \partial\Omega_{[S+1]}$ *(If SGS prec.)*
      $g_{IK} \leftarrow 0$ on each interface edge $\Gamma_{IK} \subset \partial\Omega_{[S+1]}$ *(If DS prec.)*
    Compute u_I by solving subproblem on Ω_I.
    Update transmission data for subproblems of the next group:
      $r_{JI} \leftarrow r_{JI} - r_{IJ} + 2\mathcal{B}_{IJ}u_I$ on each interface edge $\Gamma_{IJ} \subset \partial\Omega_{[S+1]}$ *(If SGS prec.)*
      $r_{JI} \leftarrow r_{JI} + 2\mathcal{B}_{IJ}u_I$ on each interface edge $\Gamma_{IJ} \subset \partial\Omega_{[S+1]}$ *(If DS prec.)*
      with J and I such that $\Omega_J \subset \Omega_{[S+1]}$ and $\Gamma_{IJ} = \Gamma_{JI}$
  end
end
// Backward sweep
for $S = N_{gr} : 2$ **do**
  parfor *each* I *such that* $\Omega_I \subset \Omega_{[S]}$ **do**
    Configure boundary data for subproblem on Ω_I:
      $u_D \leftarrow 0$ on $\partial\Omega_I \cap \Gamma_D$
      $g_{IK} \leftarrow r_{IK}$ on each interface edge $\Gamma_{IK} \not\subset \partial\Omega_{[S-1]}$
      $g_{IK} \leftarrow r_{IK}$ on each interface edge $\Gamma_{IK} \subset \partial\Omega_{[S-1]}$ *(If SGS prec.)*
      $g_{IK} \leftarrow 0$ on each interface edge $\Gamma_{IK} \subset \partial\Omega_{[S-1]}$ *(If DS prec.)*
    Compute u_I by solving subproblem on Ω_I.
    Update transmission data for subproblems of the previous group:
      $r_{JI} \leftarrow r_{JI} - r_{IJ} + 2\mathcal{B}_{IJ}u_I$ on each interface edge $\Gamma_{IJ} \subset \partial\Omega_{[S-1]}$ *(If SGS prec.)*
      $r_{JI} \leftarrow r_{JI} + 2\mathcal{B}_{IJ}u_I$ on each interface edge $\Gamma_{IJ} \subset \partial\Omega_{[S-1]}$ *(If DS prec.)*
      with J and I such that $\Omega_J \subset \Omega_{[S-1]}$ and $\Gamma_{IJ} = \Gamma_{JI}$
  end
end



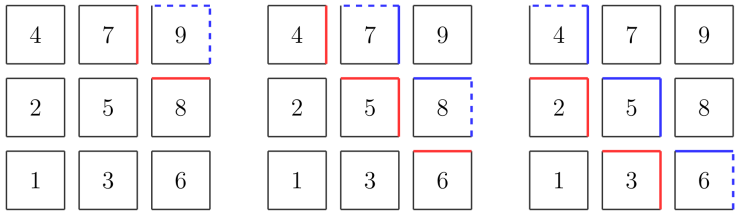
(A) The SGS preconditioner with horizontal sweeps. Resp. $\bar{\mathcal{U}}_{(3)}^{-1}, \bar{\mathcal{U}}_{(2)}^{-1}$



(B) The SGS preconditioner with diagonal sweeps. Resp. $\bar{\mathcal{U}}_{(5)}^{-1}, \bar{\mathcal{U}}_{(4)}^{-1}$ and $\bar{\mathcal{U}}_{(3)}^{-1}$



(C) The DS preconditioner with horizontal sweeps. Resp. $\bar{\mathcal{U}}_{(3)}^{-1}, \bar{\mathcal{U}}_{(2)}^{-1}$



(D) The DS preconditioner with diagonal sweeps. Resp. $\bar{\mathcal{U}}_{(5)}^{-1}, \bar{\mathcal{U}}_{(4)}^{-1}$ and $\bar{\mathcal{U}}_{(3)}^{-1}$

FIGURE 3.3: Illustration of the SGS and the DS preconditioner with horizontal sweeps and diagonal sweeps. The numbers correspond to the subdomain numbers. Red edges correspond to transmissions variables that are updated. Blue edges correspond to transmission variables that are used in the update formula.

mission variables are cancelled when solving the subdomains. Thanks to this modification, the forward and backward sweeps can be performed in parallel.

In order to illustrate the modification of the blocks of $\mathcal{L}$ and $\mathcal{U}$, we consider the 3×3 domain partition with the diagonal-type arrangement. The

blocks corresponding to the coupling of $\Omega_{[2]}$ and $\Omega_{[3]}$ read

$$\boldsymbol{\mathcal{F}}_{[3]}^{[2]} = \begin{bmatrix} \boldsymbol{\mathcal{F}}_4^2 & 0 \\ \boldsymbol{\mathcal{F}}_5^2 & \boldsymbol{\mathcal{F}}_5^3 \\ 0 & \boldsymbol{\mathcal{F}}_6^3 \end{bmatrix} = \left[\begin{array}{cccc|cccc} \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset \\ \emptyset & \mathcal{T}_4^2 & \mathcal{T}_4^2 & \mathcal{E}_4^2 & 0 & \emptyset & 0 & 0 \\ \emptyset & 0 & 0 & 0 & 0 & \emptyset & 0 & 0 \\ \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset \\ \hline \emptyset & \mathcal{T}_5^2 & \mathcal{E}_5^2 & \mathcal{T}_5^2 & 0 & \emptyset & 0 & 0 \\ \emptyset & 0 & 0 & 0 & \mathcal{T}_5^3 & \emptyset & \mathcal{T}_5^3 & \mathcal{E}_5^3 \\ \emptyset & 0 & 0 & 0 & 0 & \emptyset & 0 & 0 \\ \emptyset & 0 & 0 & 0 & 0 & \emptyset & 0 & 0 \\ \hline \emptyset & 0 & 0 & 0 & \mathcal{T}_6^3 & \emptyset & \mathcal{E}_6^3 & \mathcal{T}_6^3 \\ \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset \\ \emptyset & 0 & 0 & 0 & 0 & \emptyset & 0 & 0 \end{array}\right] \tag{3.17}$$

and

$$\boldsymbol{\mathcal{F}}_{[2]}^{[3]} = \begin{bmatrix} \boldsymbol{\mathcal{F}}_2^4 & \boldsymbol{\mathcal{F}}_2^5 & 0 \\ 0 & \boldsymbol{\mathcal{F}}_3^5 & \boldsymbol{\mathcal{F}}_3^6 \end{bmatrix} = \left[\begin{array}{cccc|cccc|cccc} \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset \\ \emptyset & 0 & 0 & \emptyset & 0 & 0 & 0 & 0 & 0 & \emptyset & \emptyset & 0 \\ \emptyset & 0 & 0 & \emptyset & \mathcal{E}_2^5 & \mathcal{T}_2^5 & \mathcal{T}_2^5 & \mathcal{T}_2^5 & 0 & \emptyset & \emptyset & 0 \\ \emptyset & \mathcal{E}_2^4 & \mathcal{T}_2^4 & \emptyset & 0 & 0 & 0 & 0 & 0 & \emptyset & \emptyset & 0 \\ \hline \emptyset & 0 & 0 & \emptyset & 0 & 0 & 0 & 0 & \emptyset & 0 & 0 & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset & \emptyset \\ \emptyset & 0 & 0 & \emptyset & 0 & 0 & 0 & 0 & \mathcal{E}_3^6 & \emptyset & \emptyset & \mathcal{T}_3^6 \\ \emptyset & 0 & 0 & \emptyset & \mathcal{T}_3^5 & \mathcal{E}_3^5 & \mathcal{T}_3^5 & \mathcal{T}_3^5 & 0 & \emptyset & \emptyset & 0 \end{array}\right], \tag{3.18}$$

where rows and columns with /0 correspond to items on boundary edges which must be removed. These blocks belong to $\boldsymbol{\mathcal{L}}$ and $\boldsymbol{\mathcal{U}}$, respectively. The modified blocks $\widetilde{\boldsymbol{\mathcal{F}}}_{[2]}^{[3]}$ and $\widetilde{\boldsymbol{\mathcal{F}}}_{[3]}^{[2]}$ are obtained by removing the terms in gray. In $\widetilde{\boldsymbol{\mathcal{F}}}_{[3]}^{[2]}$, we cancel the terms in the 3rd, 4th, 7th and 8th columns, corresponding to transmission variables on right edge and top edge of Ω_2 and Ω_3, shared with subdomains of the next group. In $\widetilde{\boldsymbol{\mathcal{F}}}_{[2]}^{[3]}$, we cancel the terms in the 2nd, 5th, 6th and 9th columns, corresponding to transmission variables on bottom edge of Ω_4, left edge and bottom edge of Ω_5 and left edge of Ω_6, shared with subdomains of the previous group. The modified blocks verify $\widetilde{\boldsymbol{\mathcal{F}}}_{[2]}^{[3]}\widetilde{\boldsymbol{\mathcal{F}}}_{[3]}^{[2]} = 0$.

3.2.3 Flexible preconditioners and parallel aspects

With both SGS and DS preconditioners, the forward and backward sweeps are performed in a direction that only depends on the blocks $-\mathcal{F}_{[I]}^{[J]}$: a horizontal direction in which the blocks in the same column are performed in parallel, and a diagonal direction in which the blocks in the same diagonal are performed in parallel. With the standard version of GMRES, only one sweeping direction must be selected. Nevertheless, in practical situations, it could be advantageous to combine different sweeping directions in order to propagate information more rapidly in different zones of the computational domain. This can be achieved thanks to the flexible version of GMRES, called F-GMRES [78, 79], where a different preconditioner can be used at each iteration. Therefore, the DS and SGS preconditioners can be used with sweeping directions that shall be modified in the course of the iterations, possibly accelerating the convergence of the iterative procedure.

The final computational procedure contains operations that can be performed simultaneously, allowing to use parallel computing architectures. The forward and backward loops are intrinsically sequential, as the different groups of subdomains have to be treated successively in a specific order. Nevertheless, parallelism can be found inside each iteration of these loops: subproblems defined on subdomains of the same group can be solved in parallel with both preconditioners. In addition, with the DS preconditioner, the forward and backward sweeps can be performed in parallel, as discussed in the previous section. For applications requiring computations with multiple right-hand sides, strategies can also be used to accelete the computations by solving all the problems in parallel instead of successively: this will be investigated in Chapter 4.

In distributed-memory parallel environments, novel questions are raised, as the placement of the subdomains on the processors shall influence the communications and the parallel efficiency. If only one subdomain is placed on each processor, all the processors will be waiting most of the time because of the sequential nature of the sweeping process. Strategies to improve the parallel efficiency have been discussed in [92] for layered-type partitions. Checkerboard partitions offer novel possibilities, as groups of subdomains can be placed on each processor, and different kind of groups can be chosen. Placement strategies have been discussed in [87] in the context of the L-sweeps preconditioners. Here, for instance, it can be advantageous to place one row of subdomains on each processor when using diagonal sweeps. This strategy could improve the parallel efficiency by reducing the waiting time of processors.

3.3 Computational results

In this section, the preconditionners are studied and compared by using several two-dimensional benchmarks solved with a high-order finite element method. We consider scattering benchmarks with a single source (section 3.3.1) and multiple sources (section 3.3.2), the Marmousi problem (section 3.3.3) and acoustic radiation from engine intake (section 3.3.4). The computational results presented in this article have been obtained with a single multicore processor. Only the solution of each subproblem was performed using shared-memory parallelism.

3.3.1 Scattering problem with a single source

We consider the scattering of a plane wave by a sound-soft circular cylinder of radius equal to 1. The scattered field is computed on a two-dimensional square domain of size 12.5×12.5, which is partitioned into an grid of 5×5 square subdomains. The scattering disk is placed in the middle of the subdomain that is at the left-down corner of the grid (first configuration) or the subdomain in the middle of the domain (second configuration). The Dirichlet boundary condition $u(\mathbf{x}) = -e^{ikx}$ is prescribed on the boundary of the disk, and the Padé-type HABC is prescribed on the exterior border of the computational domain with compatibility conditions at the corners. The Padé-type HABC operator is also used in the transmission conditions prescribed at the interfaces between the subdomains with a suited cross-points treatment (see section 2.5.3). The number of auxiliary fields $N = 8$ and the parameter $\phi = \pi/3$ are used for both exterior and transmission conditions. The wavenumber k is 2π and the wave length λ is 1.

Two numerical settings have been considered: first order (P1) triangular elements with 20 mesh vertices per wavelength (mesh elements of size $h = 1/20$), and order 7 (P7) triangular elements with 3 elements per wavelength ($h \approx 1/21$). The meshes of these two settings are made of 97868 nodes, 3966 P7 triangles and 70619 nodes, 139984 P1 triangles, respectively. The linear system resulting from the finite element discretization is solved using preconditioned versions of GMRES. Both the symmetric Gauss-Seidel (SGS) and the parallel double sweep (DS) preconditioners have been tested with three strategies:

1. Diagonal sweeps: The forward sweep goes from the left-down corner to the right-up corner of the partition, and the backward sweep does it the other way around. [SGS-D and DS-D]

2. Horizontal sweeps: The forward sweep goes from the left to the right of the partition, and the backward sweep does it the other way around. [SGS-H and DS-H]

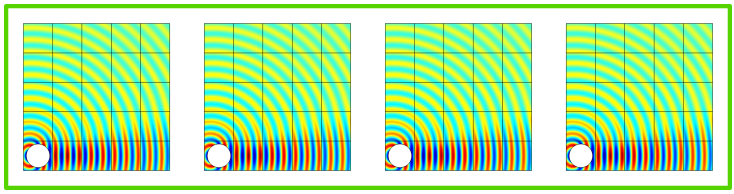
3. Two diagonal sweeping directions are used in alternance with the flexible version of GMRES (FGMRES). The sweeps are performed between the left-down and right-up corners and between the left-up and right-down corners, alternatively. [SGS-2D and DS-2D]

First configuration: Source close to the corner of the domain

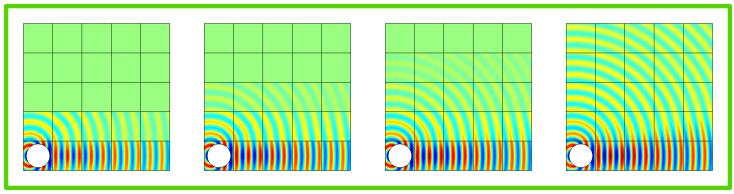
In the first configuration, the scattering disk is placed in the subdomain that is at the left-down corner of the grid. Figure 3.4 shows snapshots of the solutions after the first GMRES iterations with the different preconditioners. For all the preconditioners with diagonal sweeps, the solution is already good after only one iteration, which is due to two successful strategies for this specific case. First, the HABC operator used in the transmission conditions is particularly well-suited for scattering benchmarks. Then, the first sweep over the subdomains goes from the left-bottom corner to right-up corner, which follows the natural behavior of waves in this benchmark. Therefore, we get all correct information in all subdomains after the first iteration. For the preconditioners with horizontal sweeps, relying on round-trips between left boundary and right boundary, we can see that we get the complete solution only at fourth iteration.

The residual histories obtained with the different sweeping preconditioners are shown in Figure 3.5 for finite element schemes with P1 and P7 elements, respectively. These results confirm the visual interpretations. In both cases, the relative residual suddenly drops in residual at the first iteration when a preconditionner with diagonal sweeps is used (i.e. SGS-D, SGS-2D, DS-D and DS-2D), while it happens at the fourth iteration with horizontal sweeps (i.e. SGS-H and DS-H). Without preconditioner, eight iterations are required to reach the sudden drop in residual because the waves have to go through eight subdomains to travel from the left-down corner to the right-up corner.

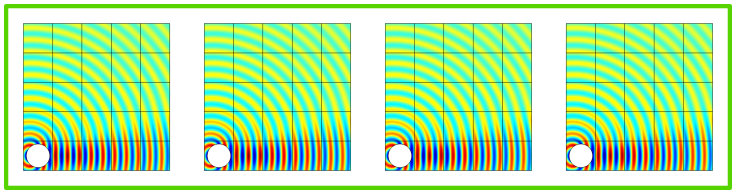
By comparing the results obtained with P1 and P7 elements, we observe that the residual histories are very similar before the sudden drop in residual with both kinds of finite elements. Nevertheless, the sudden drops are much sharper and the residuals decrease more rapidly after the sudden drop with the P7 elements. After the sudden drop in residual, we observe that the decay of the residual is nearly twice slower with the DS preconditioner than with the SGS preconditioner when diagonal sweeps is used. This must be balanced with the fact that the SGS preconditioner is intrinsically a sequential procedure, while the DS preconditioner relies on two sweeps that can be done in parallel.



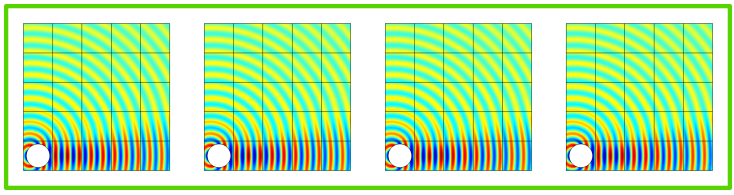
(A) SGS preconditioner with diagonal sweeps [SGS-D]



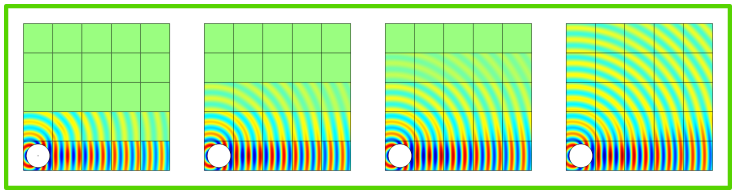
(B) SGS preconditioner with horizontal sweeps [SGS-H]



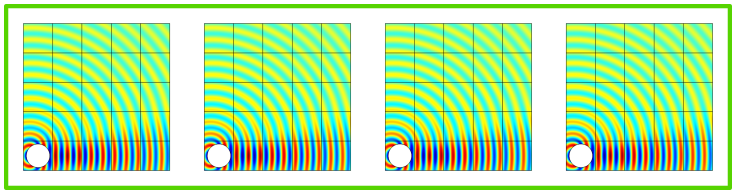
(C) SGS preconditioner with F-GMRES and alternating diagonal sweeps [SGS-2D]



(D) DS preconditioner with diagonal sweeps [DS-D]



(E) DS preconditioner with horizontal sweeps [DS-H]



(F) DS preconditioner with F-GMRES and alternating diagonal sweeps [DS-2D]

FIGURE 3.4: Scattering problem with a single source (first configuration). Snapshot of the solution after 1, 2, 3 and 4 GMRES iterations with the different preconditioners.

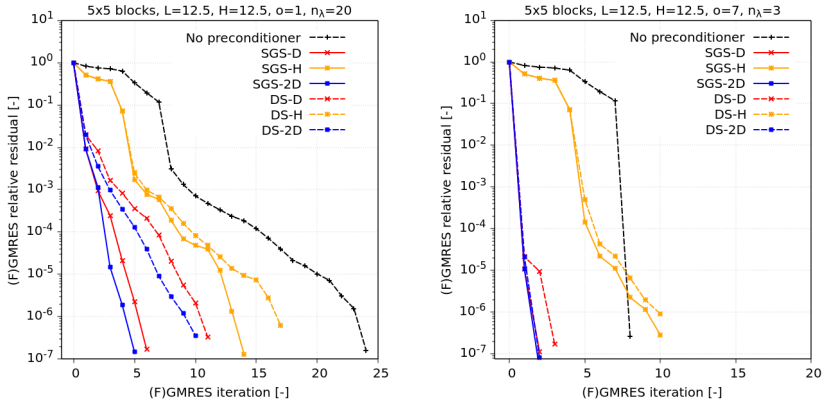
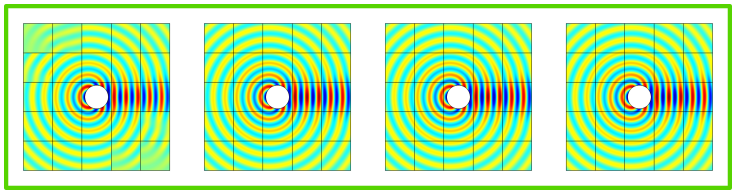


FIGURE 3.5: Scattering problem with a single source (first configuration). Residual history with/without preconditioner with P1 elements and 20 mesh vertices by wavelength (left) and with P7 elements and 3 elements by wavelength (right). Preconditioners with diagonal sweeps (SGS-D/DS-D), horizontal sweeps (SGS-H/DS-H) and alternating diagonal sweeps (SGS-2D/DS-2D) are considered.

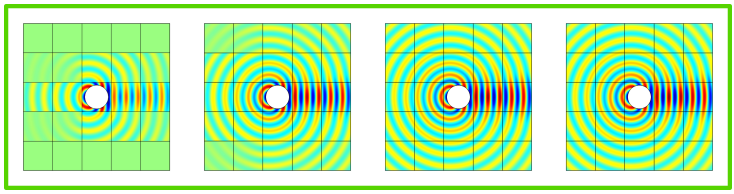
Second configuration: Source in the middle of the domain

In the second configuration, the scattering disk is placed in the middle of the computational domain. Figure 3.6 shows snapshots of the solutions after the first GMRES iterations with the different preconditioners. First, let us focus on the solutions obtained after the very first iteration. On the snapshots, we see that the zone of influence of the source corresponds to subdomains that are mainly along the diagonal direction or along the horizontal direction starting from the center of the domain. The propagation of the source in these subdomains is obviously related to the use of preconditioners with diagonal sweeps and horizontal sweeps of the subdomains, respectively.

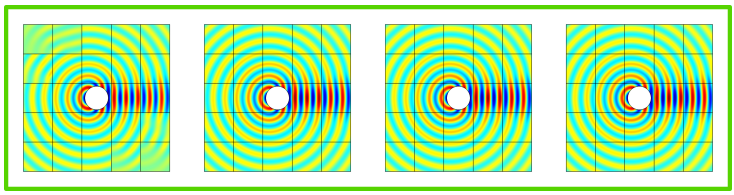
Compared to the first configuration, we have different results when the SGS and DS preconditioners are used with the diagonal sweeps or the F-GMRES strategy. Because the forward sweeps and backward sweeps of DS-D do not affect each other, the subdomains on the right-up and left-down corners cannot be reached by the source after the first iteration, which is visible on Figures 3.6 (d) and (f). By contrast, these subdomains can be reached during the backward sweep of the SGS preconditioner, which is performed after the forward sweep (Figures 3.6 (a) and (c)). With diagonal sweeps, four iterations are required to get a good solution in all the subdomains with the DS preconditioner, while only two iterations are required with the SGS preconditioner. When the strategy with F-GMRES is used, the solution is very good



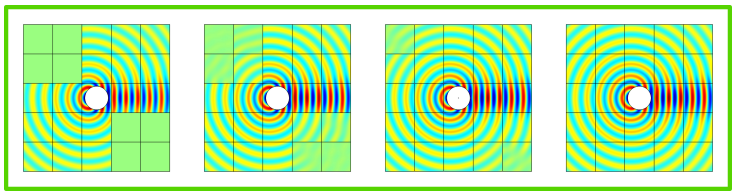
(A) SGS preconditioner with diagonal sweeps [SGS-D]



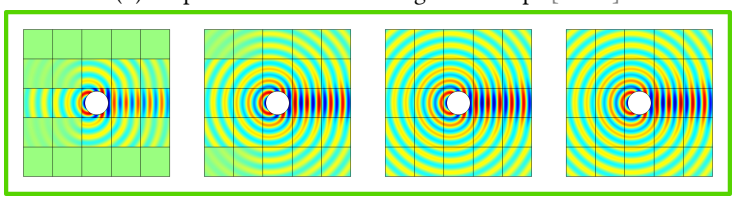
(B) SGS preconditioner with horizontal sweeps [SGS-H]



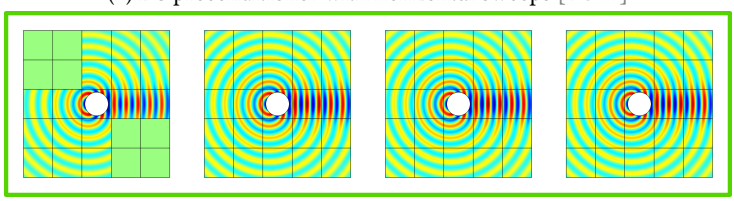
(C) SGS preconditioner with F-GMRES and alternating diagonal sweeps [SGS-2D]



(D) DS preconditioner with diagonal sweeps [DS-D]



(E) DS preconditioner with horizontal sweeps [DS-H]



(F) DS preconditioner with F-GMRES and alternating diagonal sweeps [DS-2D]

FIGURE 3.6: Scattering problem with a single source (second configuration). Snapshot of the solution after 1, 2, 3 and 4 GMRES iterations with the different preconditioners.

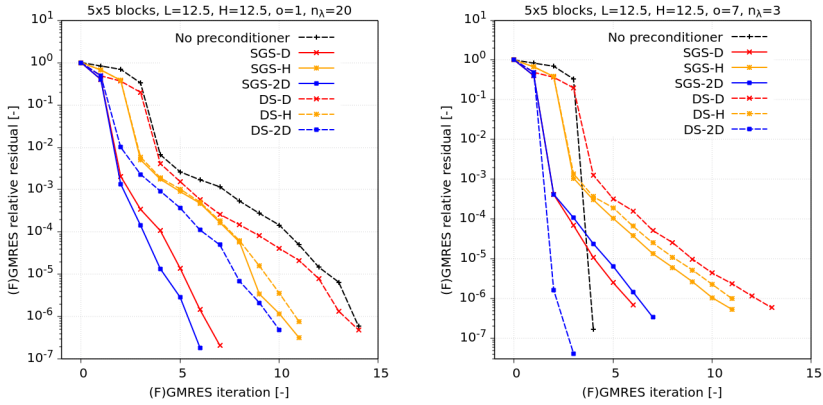


FIGURE 3.7: Scattering problem with a single source (second configuration). Residual history with/without preconditioner with P1 elements and 20 mesh vertices by wavelength (left) and with P7 elements and 3 elements by wavelength (right). Preconditioners with diagonal sweeps (SGS-D/DS-D), horizontal sweeps (SGS-H/DS-H) and alternating diagonal sweeps (SGS-2D/DS-2D) are considered.

at the second iteration, even with the DS preconditioner, because the sweeps of successive iterations are performed along both diagonal directions (Figures 3.6 (b) and (e)).

The residual histories obtained with the different sweeping preconditioners are shown in Figure 3.7 for finite element schemes with P1 and P7 elements, respectively. In all the cases, the relative residual decreases slowly until a sudden drop, which happens when the source has been propagated in all the subdomains, and when the numerical solution is close to the converged solution. The results confirm the visual observations. With the SGS preconditioner, the sudden drop occurs at the second iteration if the diagonal sweeps or the stategy with F-GMRES is used, and a third iteration is required with horizontal sweeps. With the DS preconditioner, four iterations are necessary with diagonal sweeps, while alternating diagonal sweeps realized by F-GMRES still requires two iterations.

When the source is placed in an arbitrary position in the computational domain, using flexible preconditioners, with several alternative sweeping directions, can be much more suitable than fixed preconditioners. Here, both SGS and DS preconditioners perform very well with the alternating diagonal sweeps. With P1 finite elements, the number of iterations to get the relative residual 10^{-6} is twice larger with the DS preconditioner than with the SGS preconditioner. Because the sweeps of the DS preconditioner can be done in parallel, providing a speed up of 2, both SGS and DS approaches seem equiv-

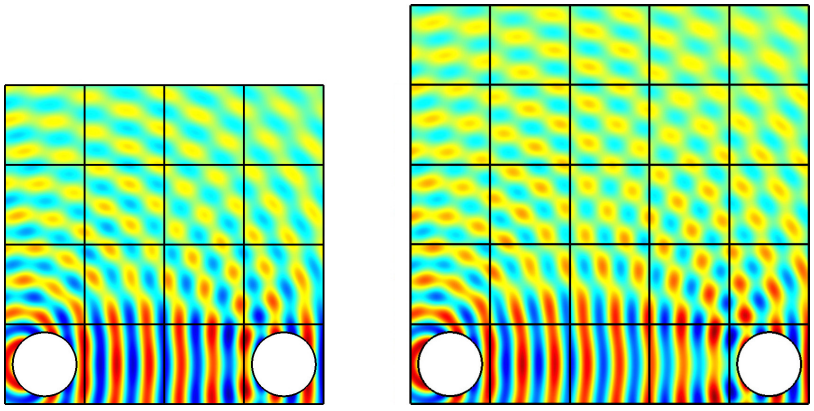


FIGURE 3.8: Scattering problem with two sources. Snapshot of the solution for configurations with 4×4 and 5×5 subdomains.

alent. By contrast, the number of iterations is similar with P7 finite elements. The parallel DS preconditioner then is more interesting for that case.

3.3.2 Scattering problem with multiple sources

We now consider the scattering of a plane wave by two sound-soft circular cylinders of unit radius. This problem is more challenging for the DD method than the previous one because the multiple reflections between both obstacles can be complicated to capture. The simulations are performed over square grids of $N_r \times N_c$ subdomains, with $N_r = N_c = 4$, 8, 12, 16 and 20. The dimension of each subdomain is 2.5×2.5. For each configuration, the scattering disks are placed at the left-down corner and the right-down corner of the grid. The physical and numerical parameters are the same as in the previous section. The numerical solutions for the 4×4 and 5×5 configurations are shown in Figure 3.8.

The numbers of GMRES iterations to reach a relative residual 10^{-6} with the different preconditioners are given in Figure 3.9 for different domain partitions. Contrary to the single obstacle case, the physical solution cannot be obtained in only one or two sweeps anymore because of the multiple reflections between both obstacles. We observe that the number of iterations increases with the number of subdomains when the preconditioners are used with a fixed horizontal or diagonal sweeps. By contrast, the strategy with the alternating diagonal sweepss keep the number of iterations constant, both with SGS and DS preconditioners. This is mainly due to the position of the disks: each of the two alternating diagonal sweeps is well suited to one of the sources. The number of iteration is slightly lower with the SGS preconditioner than with the DS preconditioner, but the latter will be more interesting

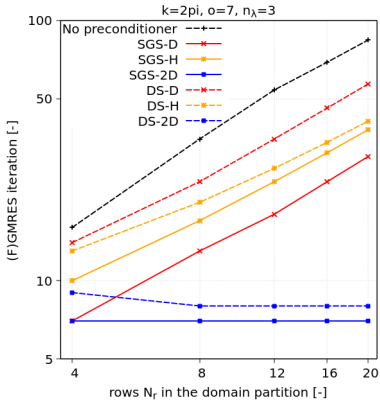


FIGURE 3.9: Scattering problem with two sources. Number of iterations with preconditioned GMRES and F-GMRES (without restart) to reach the relative residual 10^{-6} for different domain partitions.

in parallel environments considering that the forward and backward sweeps can be performed concurrently.

3.3.3 Marmousi benchmark

The Marmousi model is a 2D velocity model which is based on the geological structure of the Cuanza basin. This model exhibits a complex velocity profile $c(\mathbf{x})$ with realistic features (see Figure 3.10). It is frequently used to evaluate the performance of numerical solvers with heterogeneous media (e.g. [84]). The numerical simulations are performed over the computational domain[1] $[0, 9192] \times [0, -2904]$. The Helmholtz equation is solved over the domain, with the HABC prescribed on the boundary of the domain, and two point sources placed at coordinates $(9192/8, -10)$ and $(9192 \times 7/8, -10)$, respectively. The angular frequency is $\omega = 20\pi$, and the wavenumber is given by $k(\mathbf{x}) = \omega/c(\mathbf{x})$. Here, the maximum wavenumber is $k_{\max} = 20\pi/1500 \approx 0.042$ and the minimum wavenumber is $k_{\min} = 20\pi/4500 \approx 0.014$. In this case, the wavenumber is much smaller than 1 and the pollution term in (1) is small. Therefore, the numerical setting is only P1 triangular elements with 20 mesh vertices per wavelength. The mesh of the computational domain is made of 137808 nodes and 274018 P1 triangles. The parameters of the HABC operator are $N = 4$ and $\phi = \pi/3$ for both the exterior boundary condition and the transmission conditions prescribed at the interfaces between the subdomains. The numerical solution is shown in Figure 3.11.

The number of GMRES iterations to reach a relative residual of 10^{-6} with

[1] We assume that all the spatial dimensions are provided in the metric unit [m].

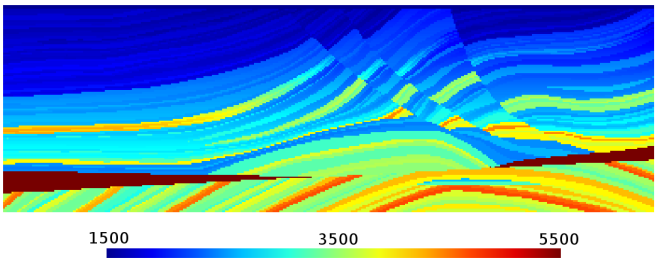


FIGURE 3.10: Marmousi benchmark. Velocity profile with values from 1500 m/s to 5500 m/s.

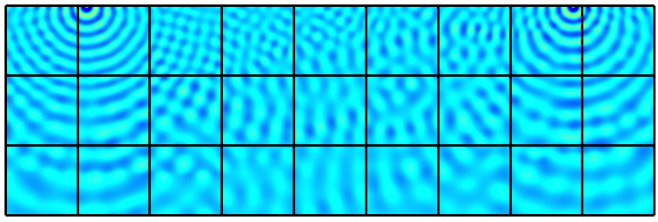


FIGURE 3.11: Marmousi benchmark. Numerical solution and domain partition with 3×9 subdomains.

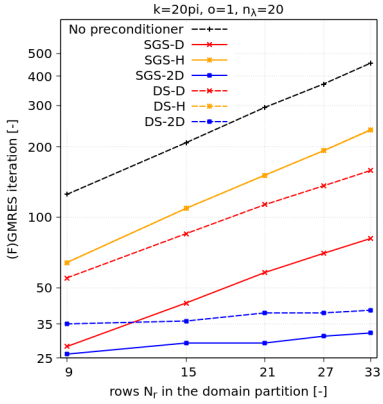


FIGURE 3.12: Marmousi benchmark. Number of iterations with preconditioned GMRES and F-GMRES (without restart) to reach the relative residual 10^{-6} for different domain partitions. The domain is partitioned into $N_r \times N_c$ subdomains, with $N_r = 9, 15, 21, 27, 33$ and $N_c = 3N_r$.

the different preconditioners is given in Figure 3.12. We have considered domain decompositions into rectangular grids of $N_r \times N_c$ subdomains, where the number of rows is $N_r = 3, 9, 15, 21, 27, 33$ and the number of columns

is $N_c = 3N_r$. We observe that the number of iterations increases with the number of subdomains with all the preconditionners, but the increase is very slown with F-GMRES and the switching sweeping directions. Between the coarsest domain partition (3×9 subdomains) and the finest partition (33×99 subdomains), the number of iterations has increased by a factor between 5 and 9 with the preconditioners with fixed sweeping directions (SGS-D, SGS-H, DS-D, DS-H), while the factor is smaller than 2 with the switching sweeping directions (SGS-2D, DS-2D). In nearly all the cases, the SGS preconditioner with switching sweeping directions requires the smallest number of iterations, but, considering the possible parallelisation of the forward/backward sweeps with the DS preconditioner, that strategy is the best if a parallel environment is used.

3.3.4 Acoustic radiation from engine intake

Description of the benchmark and domain partition

In this last benchmark, we address the computation of a time-harmonic acoustic field in a computational domain that is not rectangular. It deals with the aeroacoustics of an idealized turbofan engine intake. The geometry, shown in Figure 3.13, is a cylindrical duct of slowly-varying cross-section. The 2D Helmholtz equation is solved on this computational domain, which is included inside the rectangular region $[-0.3, 3.0] \times [0.0, 2.5]$. We consider a Dirichlet condition on the source, $u|_{\text{source}} = \sin((2\pi/0.50) \cdot y)$, homogeneous Neumann boundary condition on the hardwalls, and an HABC on the artificial borders where the waves must be radiated (see Figure 3.13). For the numerical solution, P5 finite elements are used with 5 elements per wavelength. The mesh of the computational domain is made of about 3.5×10^7 nodes and 2.8×10^6 P5 triangles for wavenumber 160π ($h \approx 1/2000$). The parameters of the HABC operators are $N = 4$ and $\phi = \pi/3$ for both interface and exterior edges. The numerical solution corresponding to wavenumber $k = 160\pi$ is shown on Figure 3.14.

Because the domain is not rectangular, additional steps are required to apply the proposed sweeping preconditioners, which are designed a priori only for checkerboard partitions. We have generated domain partitions of the rectangular region that contains the computational domain. This process is performed with Gmsh before the mesh generation. Then, every partition contains $N_r \times N_c$ rectangular subdomains (see Figure 3.15), but several "null" subdomains are fully outside the computational domain (e.g. subdomain Ω_J on Figure 3.15, right), and several subdomains are crossed by the border of the computational domain (e.g. subdomains Ω_I, Ω_L and Ω_K on Figure 3.15, right). After discretization, no unknown is associated to the null subdomains. The number of unknowns is smaller for subdomains that are crossed by the

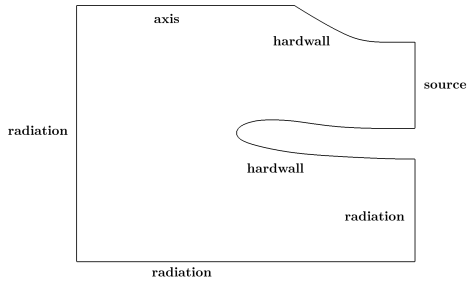


FIGURE 3.13: Benchmark *"Engine intake"*. Representation of the computational domain and boundary conditions.

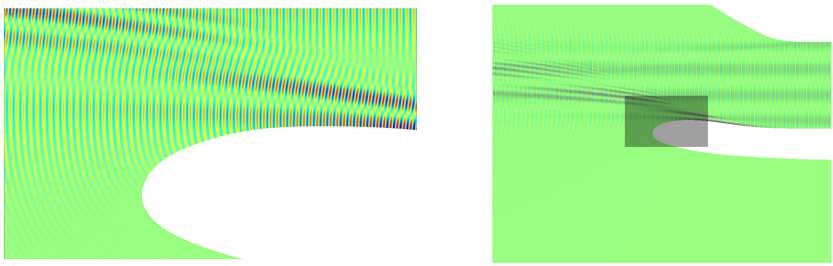


FIGURE 3.14: Benchmark *"Engine intake"*. Snapshot of the numerical solution for wavenumber $k = 160\pi$.

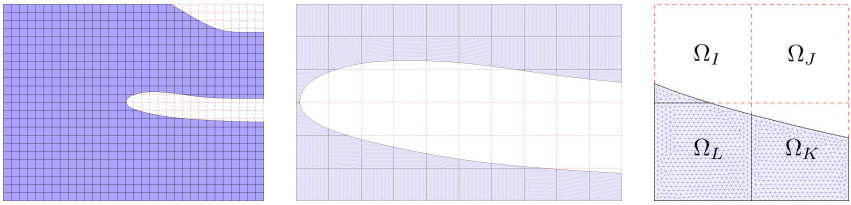
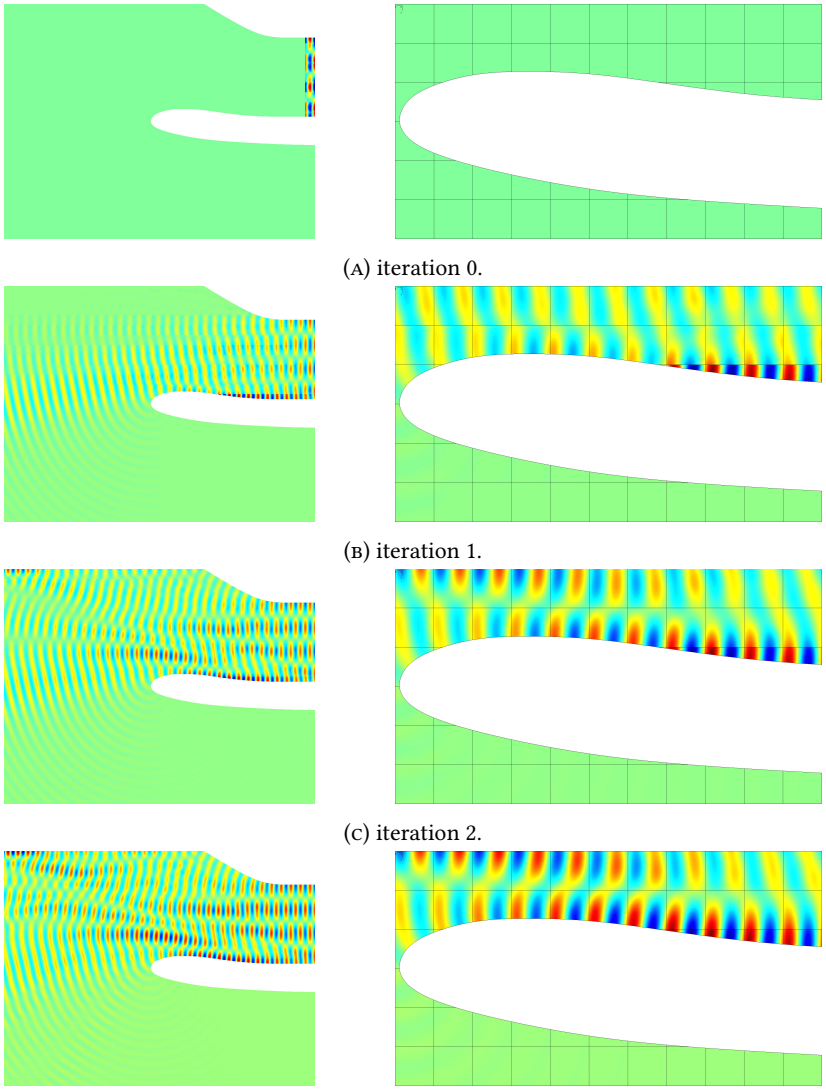


FIGURE 3.15: Benchmark *"Engine intake"*. Mesh and example of domain partition.

domain border than for rectangular subdomains that are fully contained inside the computational domain. Similarily, the number of discrete transmission variables is smaller on interface edges crossed by the domain boundary. In practice, the solution procedure is performed by iterating over all the subdomains, wathever they are inside or outside the computational domain. Dummy systems and dummy vectors of variables are associated to the null subdomains, and dummy transmission variables are associated to interface edges which do not belong to the computational domain (i.e. red dashed edges on Figure 3.15). Therefore, the sweeping preconditioner and our computational code can be straightforwardly used for this benchmark as well.



(A) iteration 0.

(B) iteration 1.

(C) iteration 2.

(D) iteration 3.

FIGURE 3.16: Benchmark *"Engine intake"*. Snapshot of the solution at the beginning of the procedure (iteration 0) and after 1, 2 and 3 iterations with SGS-2D. The wavenumber is 80π.

Results

Figure 3.16 presents the snapshot of the solution with 48×36 subdomains at the beginning of the procedure from iteration 0 to iteration 3 with preconditioners SGS-2D. The wavenumber k is 80π in Figure 3.16 instead of 160π considering that the visualisation is more clearly visible. At the first iteration,

(A) Number of iterations (B) Runtime in seconds

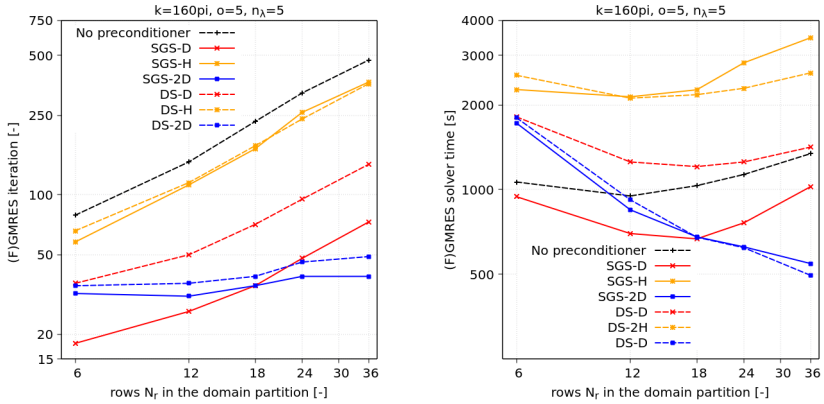


FIGURE 3.17: Benchmark *"Engine intake"*. Number of iterations (a) and runtime in seconds (b) with preconditioned GMRES and F-GMRES (with restart) to reach the relative residual 10^{-6} for different domain partitions. The domain is partitioned into $N_r \times N_c$ subdomains, with $N_r = 6$, 12, 18, 24, 36 and $N_c = 4N_r/3$.

partial information is obtained by sweeps that go from the left-down corner to the right-up corner and the other is missed at the top of the computational domain. At the second iteration, the information at the top of the computational domain has been added, which is contributed by alternating sweeps. At the third iteration, additional information is obtained in the computational domain.

The number of iterations and the runtime to reach a relative residual of 10^{-6} with the different preconditioners are given in Figure 3.17. Simulations are carried out on the Intel Xeon Phi (CPU 7210@1.30GHz). The runtime corresponds to the restarted (F)GMRES resolution phase (number of restart = 20). The number of threads is equal to the number of rows of subdomains.

The convergence rate with SGS-2D is the fastest. The results are comparable with DS-2D. With both preconditioners, the number of iterations is stable.

Comparing flexible preconditioners and fixed preconditioners, we can also see that switching preconditioners improves the robustness of the DD method. SGS-H and DS-H are not good enough, as we can see that the number of iterations with both preconditioners increases with the number of rows of subdomains.

Comparing all the solvers, the DD methods with switching preconditioners perform best. Indeed, when the number of subdomains increases, the runtime is significantly smaller with switching preconditioners, which is di-

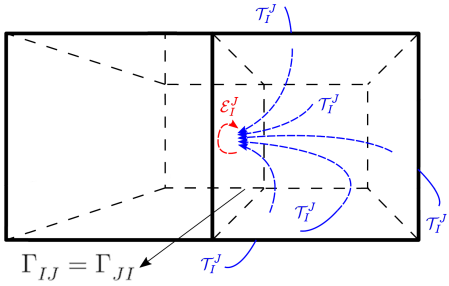


FIGURE 3.18: Illustration of the coupling introduced by the self-coupling operator (in red) and the transfer operator (in blue) for a configuration with neighboring subdomains Ω_I and Ω_J with the shared interface $\Gamma_{IJ} = \Gamma_{JI}$.

rectly linked with the smaller number of iterations they need to converge. While SGS-D or DS-D also reduce the number of iterations, the inner steps of these preconditioners at each iteration are more time-consuming, so that overall they don't lead to significant improvement on computation times.

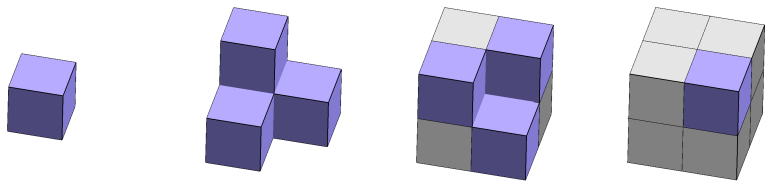
3.4 Extension to three dimensions

3.4.1 Algebraic structure of the interface problem

The global interface problem in three dimensions can be written in exactly the same way as in two dimensions, i.e. as (3.1). The global matrix $\boldsymbol{\mathcal{F}} := \boldsymbol{\mathcal{I}} - \boldsymbol{\mathcal{A}}$ is still constituted by $N_{\text{dom}} \times N_{\text{dom}}$ blocks, each corresponding to the coupling between the unknowns of two subdomains Ω_I and Ω_J ($I, J = 1 \dots N_{\text{dom}}$ and $I \neq J$). In three dimensions, the interfaces of one subdomain are 6 faces if there is no exterior boundaries, which is different from two-dimensional cases in which there are 4 shared edges for one subdomain.

3.4.1.1 Identification of the blocks

Using the block representation, the abstract system (3.1) in three dimensions can also be rewritten as equation (3.2), where the vectors $\mathbf{g}_I$ and $\mathbf{b}_I$ contain all the transmission variables and the source terms of six faces and related variables on edges and at corners, respectively, for the cubic subdomain Ω_I. The description of the identification of the blocks in the three-dimensional case is similar to that in the two-dimensional case. The only difference is that the blocks $\mathcal{F}_I^J$, $\mathbf{g}_J$, and $\mathbf{b}_I$ are of size six. Moreover, since there are at most six neighbouring subdomains for each subdomains, there are at most six



(A) 1st group (B) 2nd group (C) 3rd group (D) 4th group

FIGURE 3.19: Illustration of groups of subdomain in three dimensions

off-diagonal blocks in each line and each column of the global block matrix. Considering that the difference is trivial, we don't repeat the decomposition of Equation (2.13), the self-coupling and transfer operators in three-dimensional cases and we directly present the block $\mathcal{F}_I^J$. For instance, assuming the the shared interface edge is $\Gamma_{IJ} = \Gamma_{JI}$ (see Figure 3.18), the block reads

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ g_{IJ} \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \mathcal{E}_I^J & \mathcal{T}_I^J & \mathcal{T}_I^J & \mathcal{T}_I^J & \mathcal{T}_I^J & \mathcal{T}_I^J \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{g}_J = \begin{bmatrix} 0 \\ 0 \\ 0 \\ g_{IJ} \\ 0 \\ 0 \end{bmatrix} + \mathcal{F}_I^J \mathbf{g}_J = \mathbf{b}_I, \quad (3.19)$$

where the vector $\mathbf{g}_J$ contains all the transmission variables for the six faces which are in order: left, bottom, back, right, top, and front. This example corresponds to the illustration in Figure 3.18. In the general case with subdomains touching the exterior border of the main domain, corresponding self-coupling operator and transfer operators are cancelled.

3.4.1.2 Block matrix forms for the global system

The sparse structure of the global block matrix in three dimensions can also be re-formed into a tridiagonal matrix. With Rubik's-cube-type partitions, the subdomains should also be arranged into diagonal groups. These groups are illustrated in Figure 3.19 for a $2 \times 2 \times 2$ Rubik's-cube-type decomposition.

With the diagonal-type arrangement, the global system can be written as

$$\left[\begin{array}{cccccccc}
\mathbf{I}_1 & \mathcal{F}_1^2 & \mathcal{F}_1^3 & \mathcal{F}_1^5 & & & & \\
\mathcal{F}_2^1 & \mathbf{I}_2 & & & \mathcal{F}_2^4 & \mathcal{F}_2^6 & & \\
\mathcal{F}_3^1 & & \mathbf{I}_3 & & \mathcal{F}_3^4 & & \mathcal{F}_3^7 & \\
\mathcal{F}_5^1 & & & \mathbf{I}_5 & & \mathcal{F}_5^6 & \mathcal{F}_5^7 & \\
& \mathcal{F}_4^2 & \mathcal{F}_4^3 & & \mathbf{I}_4 & & & \mathcal{F}_4^8 \\
& \mathcal{F}_6^2 & & \mathcal{F}_6^5 & & \mathbf{I}_6 & & \mathcal{F}_6^8 \\
& & \mathcal{F}_7^3 & \mathcal{F}_7^5 & & & \mathbf{I}_7 & \mathcal{F}_7^8 \\
& & & & \mathcal{F}_8^4 & \mathcal{F}_8^5 & \mathcal{F}_8^6 & \mathbf{I}_8
\end{array}\right]
\left[\begin{array}{c} \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \\ \mathbf{g}_5 \\ \mathbf{g}_4 \\ \mathbf{g}_6 \\ \mathbf{g}_7 \\ \mathbf{g}_8 \end{array}\right]
=
\left[\begin{array}{c} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \\ \mathbf{b}_5 \\ \mathbf{b}_4 \\ \mathbf{b}_6 \\ \mathbf{b}_7 \\ \mathbf{b}_8 \end{array}\right], \tag{3.20}$$

where $\mathbf{I}_I$ is the identity matrix associated to the subdomain Ω_I. The matrix of the system can be written as a block tridiagonal matrix with large blocks corresponding to interactions between two groups of subdomains. Here, each group corresponds to the subdomains on a given diagonal of the domain partition (see Figure 3.20). In the matrix, the large diagonal blocks are identity matrices because subdomains belonging to the same group are never neighbours. The large off-diagonal blocks are rectangular with different sizes because the groups contain different numbers of subdomains. For a general $N_x \times N_y \times N_z$ decomposition of the domain, there are $N_x + N_y + N_z - 2$ groups of subdomains and the matrix of the system can be still be written as a block tridiagonal matrix.

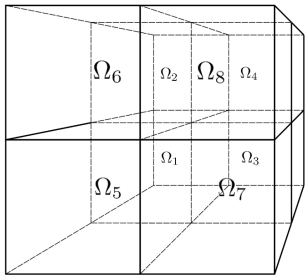
For convenience, we introduce the general representation

$$\left[\begin{array}{cccc}
\mathcal{F}_{[1]}^{[1]} & \mathcal{F}_{[1]}^{[2]} & & \\
\mathcal{F}_{[2]}^{[1]} & \mathcal{F}_{[2]}^{[2]} & \mathcal{F}_{[2]}^{[3]} & \\
& \mathcal{F}_{[3]}^{[2]} & \mathcal{F}_{[3]}^{[3]} & \mathcal{F}_{[3]}^{[4]} \\
& & \mathcal{F}_{[4]}^{[3]} & \mathcal{F}_{[4]}^{[4]}
\end{array}\right]
\left[\begin{array}{c} \mathbf{g}_{[1]} \\ \mathbf{g}_{[2]} \\ \mathbf{g}_{[3]} \\ \mathbf{g}_{[4]} \end{array}\right]
=
\left[\begin{array}{c} \mathbf{b}_{[1]} \\ \mathbf{b}_{[2]} \\ \mathbf{b}_{[3]} \\ \mathbf{b}_{[4]} \end{array}\right], \tag{3.21}$$

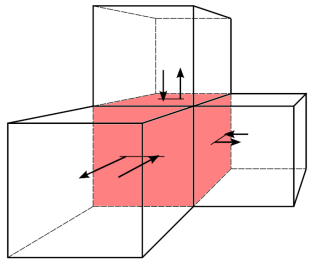
where the vectors $\mathbf{g}_{[I]}$ and $\mathbf{b}_{[I]}$ are associated to one group of subdomains and each block $\mathcal{F}_{[I]}^{[J]}$ corresponds to the coupling between the transmission variables of two groups. Each block corresponds to one box in (3.20).

3.4.2 Preconditioned parallel solvers

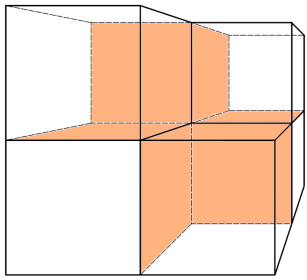
According to the block matrix forms for the global system in two and three dimension, we can see that the structure of (3.13) and (3.21) are same. There-



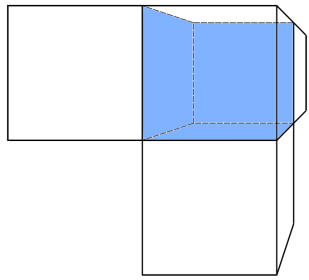
(A) numerate 1^{st}, 2^{nd}, $\cdots$, 8^{th} subdomains



(B) Operators between 1^{st} and 2^{nd} group



(C) Operators between 2^{nd} and 3^{rd} group



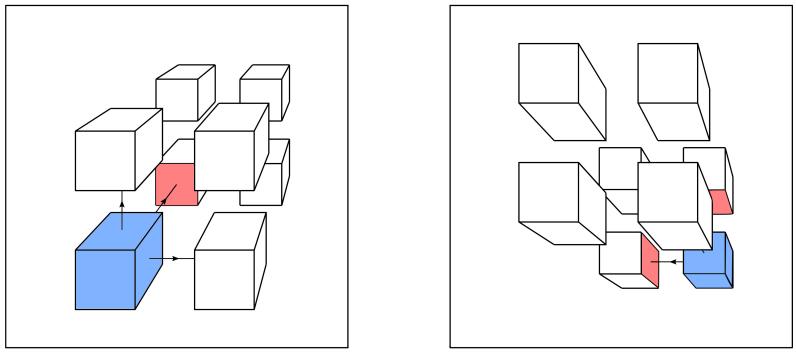
(D) Operators between 3^{rd} and 4^{th} group

FIGURE 3.20: Illustration of operators $\mathcal{F}_J^I$ in the global structure.

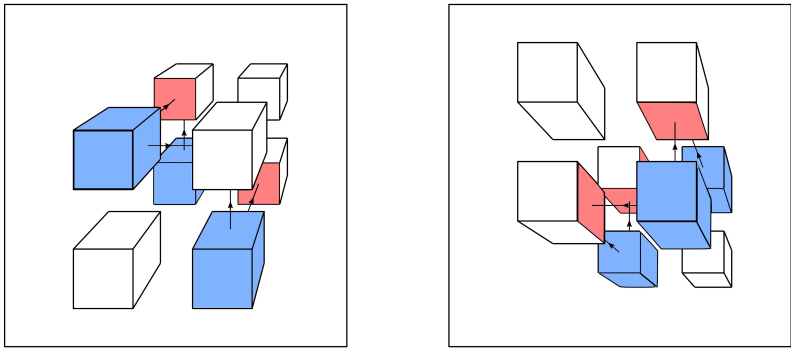
fore, there is no difficulty to extend preconditioned parallel solvers from 2D to 3D.

3.4.2.1 Block Symmetric Gauss-Seidel (SGS) preconditioning

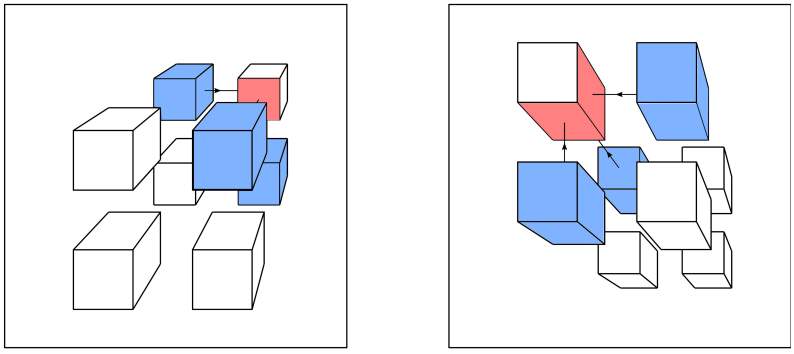
The general block Symmetric Gauss-Seidel (SGS) preconditioner presented in section 3.2.1 can be applied in 3D as well. Comparing to the 2D case, forward sweeps and backward sweeps propagate information from one subdomain to three neigbouring subdomains across surfaces (see Figure 3.21). As we have seen in 2D that it is more efficient to apply preconditioners with a diagonal-type arrangement, we only present this case in 3D. For each iteration of the loops for the 3D case, the solution of the subproblems of a given group can also be performed in parallel. In Figure 3.21, we can see that there is no cou-



(A) Forward sweep of SGS preconditioner from group 1 to group 2. Blue faces indicate configuration of boundary data for subproblems in group 1. Red faces indicate update of transmission data for subproblems in group 2.



(B) Forward sweep of SGS preconditioner from group 2 to group 3. Blue faces indicate configuration of boundary data for subproblems in group 2. Red faces indicate update of transmission data for subproblems in group 3.



(C) Forward sweep of SGS preconditioner from group 3 to group 4. Blue faces indicate configuration of boundary data for subproblems in group 3. Red faces indicate update of transmission data for subproblems in group 4.

FIGURE 3.21: Illustration of SGS preconditioners, in left pictures, we observe from the positive direction of $x-$axis, and in right pirctures, we observe from the negative direction of $x-$axis.

pling between the subdomains of the same group since there is no shared faces between them. The forward and backward sweeps must be performed sequentially.

3.4.2.2 Double Sweep (DS) preconditioning

With the DS preconditioning, the $\mathcal{L}$ and $\mathcal{U}$ matrices are modified in such a way that the application of one of them do not modify the elements of the vector used by the other. We can write $\widetilde{\mathcal{F}}_{\mathrm{DS}} = \widetilde{\mathcal{L}}\,\widetilde{\mathcal{U}} = \widetilde{\mathcal{U}}\,\widetilde{\mathcal{L}} = \widetilde{\mathcal{L}} + \widetilde{\mathcal{U}} - \mathcal{I}$. The forward and backward step can then be performed in parallel, which potentially reduces the runtime per iteration by a factor two.

To remove the dependencies between $\mathcal{L}$ and $\mathcal{U}$, the blocks $\mathcal{F}_{[I]}^{[J]}$ are modified in such a way that, for a given group of subdomains,

- the forward sweep does not use transmission data from faces shared with the next group of subdomains (these data are modified by the backward sweep);
- the backward sweep does not use transmission data from faces shared with the next group of subdomains (these data are modified by the forward sweep).

The effective update process is illustrated in Figure 3.22. For each iteration of the forward sweep, transmission data defined on the blue faces are used to update the transmission data on the red faces for the next subdomain.

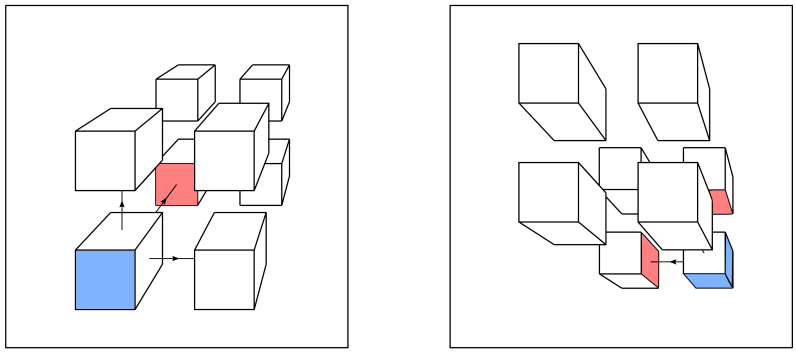
To ensure $\widetilde{\mathcal{L}}\,\widetilde{\mathcal{U}} = \widetilde{\mathcal{L}} + \widetilde{\mathcal{U}} - \mathcal{I}$, the blocks must be modified such a way that $\widetilde{\mathcal{F}}_{[I]}^{[J]}\,\widetilde{\mathcal{F}}_{[J]}^{[I]} = 0$:

- In $\mathcal{F}_{[J]}^{[I]}$, we cancel the terms corresponding to transmission variables of boundaries which are shared with subdomains of the *'next'* group.
- In $\mathcal{F}_{[I]}^{[J]}$, we cancel the terms corresponding to transmission variables of boundaries which are shared with subdomains of the *'previous'* group.

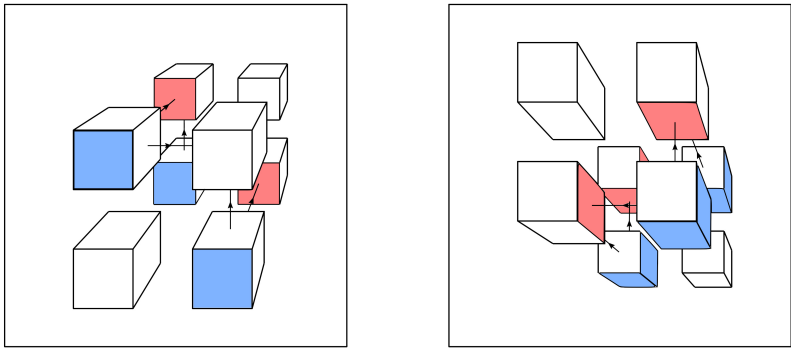
In pratice, the computational procedure is very close to the SGS preconditioning. When solving the subproblems, one or more transmission variables are cancelled (see Algorithm 5). In each group, the subproblems can still be solved in parallel.

3.4.3 GMRES and flexible GMRES

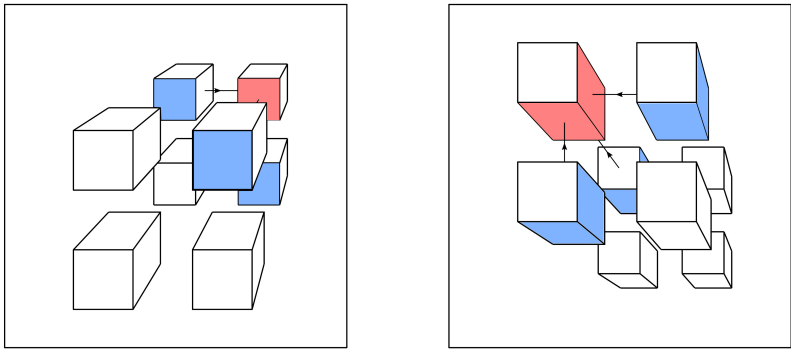
In the 3D case, in order to sweep all subdomains, it is necessary to sweep in four directions at least (see Figure 3.23). Besides this change, the same procedure is applied using the flexible variant of GMRES, GMRES. For parallel computing with distributed memory, it is important to consider the assignment



(A) Forward sweep of DS preconditioner from group 1 to group 2. Blue faces indicate configuration of boundary data for subproblems in group 1. Red faces indicate update of transmission data for subproblems in group 2.

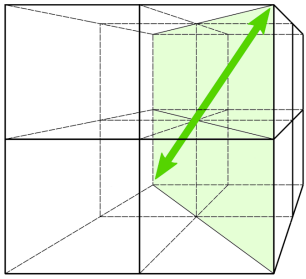


(B) Forward sweep of DS preconditioner from group 2 to group 3. Blue faces indicate configuration of boundary data for subproblems in group 2. Red faces indicate update of transmission data for subproblems in group 3.

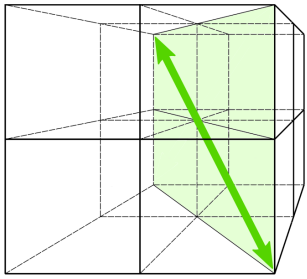


(C) Forward sweep of DS preconditioner from group 3 to group 4. Blue faces indicate configuration of boundary data for subproblems in group 3. Red faces indicate update of transmission data for subproblems in group 4.

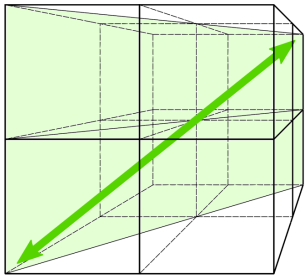
FIGURE 3.22: Illustration of DS preconditioners, in left pictures, we observe from the positive direction of $x-$axis, and in right pirctures, we observe from the negative direction of $x-$axis.



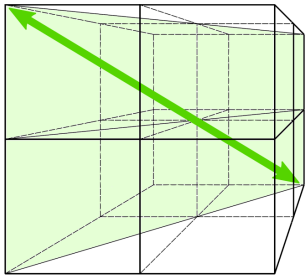
(A) Type 1 of Round trips.



(B) Type 2 of Round trips.



(C) Type 3 of Round trips.



(D) Type 4 of Round trips.

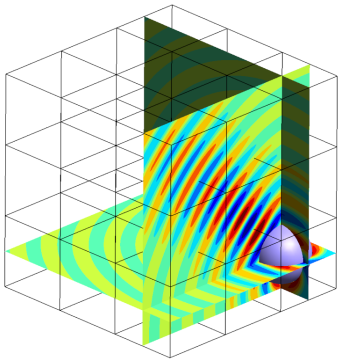
FIGURE 3.23: Round trips for diagonal-type preconditioners in 3D.

of subdomains to the processors. Compared to the 2D case, the assignments in 3D are done as follows:

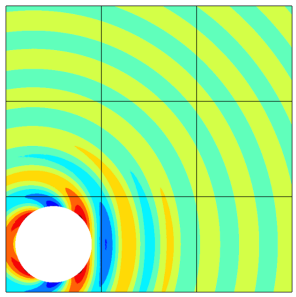
- If there are $N_x \times N_y$ processors, every processor deals with the N_z subdomains which are adjacent to each other in the z-direction;
- If there are $N_x \times N_y \times N_z$ processors, each processor deals with one subdomain.

3.4.4 Benchmarks

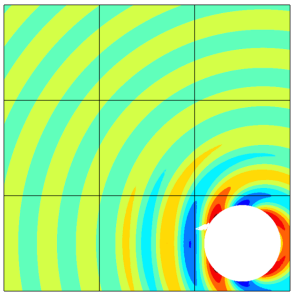
We consider the scattering of a plane wave by a sound-soft sphere of radius equal to 1. The scattered field is computed in a three-dimensional cube domain of size $7.5 \times 7.5 \times 7.5$, which is partitioned into $3 \times 3 \times 3$ cube subdomains. The scattering sphere is placed in the middle of the subdomain that is nearest to the origin. The Dirichlet boundary condition $u(\mathbf{x}) = -e^{i\boldsymbol{\kappa}\cdot\mathbf{x}}$ is prescribed on the boundary of the sphere, and the Padé-type HABC is prescribed on the exterior boundary and the artificial interfaces, with compatibility conditions on the edges and at the corners. The number of auxiliary fields $N = 4$ and



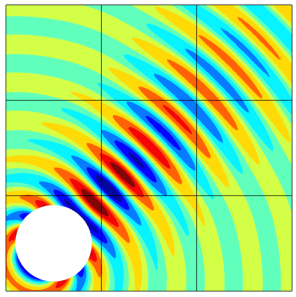
(A) General visualisation.



(B) +X view.



(C) +Y view.



(D) +Z view.

FIGURE 3.24: Three-dimensional scattering problem. Numerical solution and domain partition with $3 \times 3 \times 3$ subdomains.

the parameter $\phi = \pi/3$ are used for both exterior and interior transmission conditions. The wavenumber $||\boldsymbol{\kappa}||$ is equal 2π with direction $[1/\sqrt{2}, 1/\sqrt{2}, 0]$. We use a tetrahedral mesh with 3 $P7$ finite elements per wavelength, resulting in about 4 million nodes.

For all the preconditioners with diagonal sweeps, the solution is already good after only one iteration, which is similar to what as observed in the two-dimensional case. For the preconditioners with horizontal sweeps, we get a relatively good solution at the fourth iteration, after a sudden drop of the residual. The explainations are the same as in the 2D setting: the HABC operator used on the interfaces is particularly well-suited for this simple scattering problem, and therefore we get directly all correct information in all subdomains once the information has been transmitted to the exterior boundary of the domain. Compared to the 2D cases, the residual decreases more slowly after the initial drop, which might be due to the worse condition number of

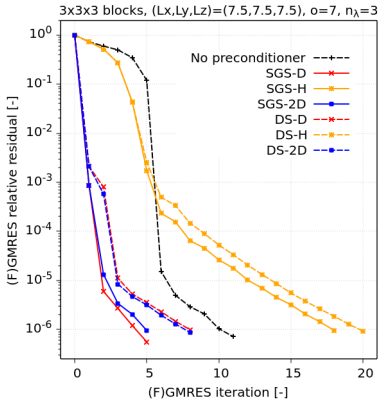


FIGURE 3.25: Scattering problem with a single source at a corner in three dimensions. Residual history with/without preconditioner with P7 elements and 3 elements by wavelength. Preconditioners with diagonal sweeps (SGS-D/DS-D), horizontal sweeps (SGS-H/DS-H) and alternating diagonal sweeps (SGS-2D/DS-2D) are considered.

the operator in 3D.

3.5 Conclusion

In this chapter we have proposed and compared multidirectional sweeping preconditioners for the finite element solution of Helmholtz problems with block-type domain decompositions. These family of preconditioner constitute a generalization of the sweeping techniques that have previously been proposed for layered domain partitions, and have the advantage to present much better parallel efficiency. Horizontal, vertical and diagonal sweeping directions can be used with symmetric Gauss-Seidel and parallel double sweep preconditioners, and several directions can be combined by using the flexible version of GMRES. We have shown on several applicative cases that these preconditioners provide an efficient way to rapidly transfer information in the different zones of the computational domain, thus dramatically accelerating the convergence of the iterative solution procedures with GMRES. We have observed that the diagonal sweeping directions, with flipping between each iteration of the flexible GMRES, were particularly efficient in all the cases.

In the next chapter we investigate how these preconditioned DD methods can be modified to handle multiple right-hand sides, a particularly important feature for applications e.g. in imaging.

Improved parallelization strategy for multiple right-hand sides

4

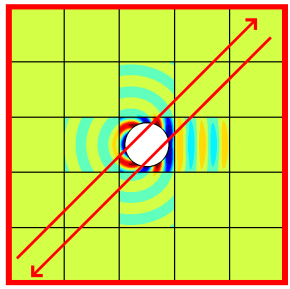
Seismic imaging is a geophysical technique that allows to investigates the subsurface of the earth. The idea of seismic imaging is to direct an high level sound source in the direction of the ground. Receivers called geophones, analogous to microphones, pick up the echoes that come back up through the ground and record echo signals. This forward part of the process is done in the field using either a dedicated ship or airplane.

Inverse imaging allows to turn these signals into images of the geologic structure of the underground, with the aim of eventually discovering a hydrocarbon deposit. This part is done on a computer and is formulated as an inverse problem: finding velocities $c(\mathbf{x})$ in Helmholtz equation that match the echo signals. Technically, this requires to solve a given Helmholtz problem with multiple right-hand sides.

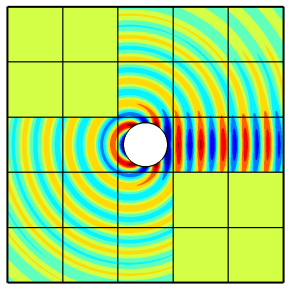
In the previous chapter we have presented a family of generalized sweeping preconditioners. At that point, we were considering one single right-hand side. In this final chapter, we propose an improved parallelization strategy for the generalized sweeping preconditioners, which focuses on the cases of multiple right-hand sides.

4.1 Introduction

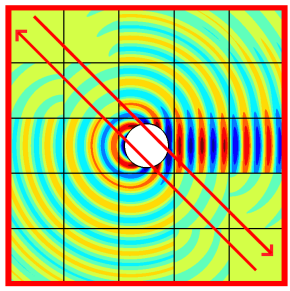
In [92], A. Vion and C. Geuzaine designed additional parallelization strategies in order to overcome the sequential nature of the sweeping process with layered-type domain decompositions. The main idea of these strategies is to execute several sweeping preconditioners simultaneously on different right-hand sides in a pipelining fashion. However, these strategies cannot reach a perfect parallel scaling: the proposed technique starts by solving one single right hand side, then two and so forth. In order to overcome this drawback, [92] introduces sweeping preconditioners with cuts that maximize the usage of computers resources for multiple right-hand sides. Yet, those cuts involve a new issue: the more cuts we have, the more iterations we need to converge.



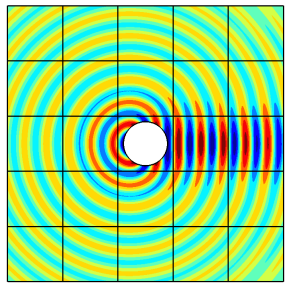
(A) Iteration 1, the application of $\mathcal{F}$



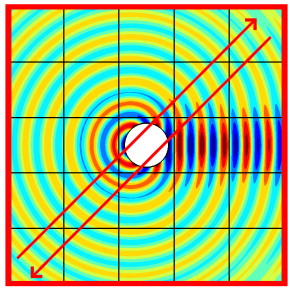
(B) Iteration 1, the application of $\mathcal{F}^{-1}$



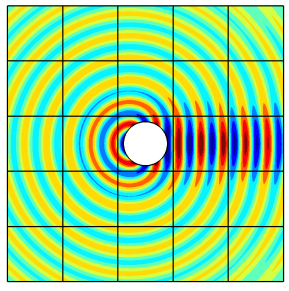
(C) Iteration 2, the application of $\mathcal{F}$



(D) Iteration 2, the application of $\mathcal{F}^{-1}$



(E) Iteration 3, the application of $\mathcal{F}$



(F) Iteration 3, the application of $\mathcal{F}^{-1}$

FIGURE 4.1: Visualisation: full sweeping preconditioners.

In this final chapter, we start from [92] and modify the approach of sweeping preconditioners with cuts. Here, we only use one cut in sweeping preconditioners. In this new strategy, the sequence of *steps* in sweeping preconditioners with one cut for one right-hand side is the key ingredient to reach a faster rate of convergence and to maximize the usage of resources.

4.2 Full sweeping preconditioners

For the sake of simplicity, we only consider double sweep sweeping preconditioners with the diagonal-type arrangement. Full sweeping preconditioners

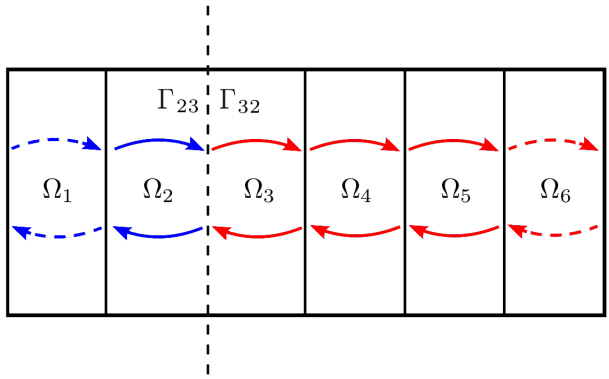


FIGURE 4.2: Groups are cut by the dashed line. The blue arrows show forward and backward steps in left box and the red arrows in right box. There are no steps for dashed arrows. The dashed arrows are placed for keeping the structure clear.

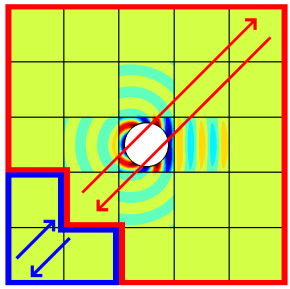
sweep group by group forward and backward. Let's recall some notations of the previous Chapter: the global system is represented by Equation (3.13), where transimission matrix $\mathcal{F}_{[i]}^{[i+1]}$ stands for one *backward step* from $i+1^{\text{th}}$ group to i^{th} group and transimission matrix $\mathcal{F}_{[i+1]}^{[i]}$ stands for one *forward step* from i^{th} group to $i+1^{\text{th}}$ group. In order to understand what follows, it is interesting to present visually how the global system (3.13) and the DS preconditioner transmit information in the case of the scattering of a plane wave by a sound-soft circular cylinder (see Figure 4.1):

- Figure 4.1a shows the wave propagation at the 1^{st} iteration of the iterative solution of the global Schwarz system, with red arrows indicating the coming application, i.e. the preconditioning;
- Figure 4.1b shows the wave propagation at the 1^{st} iteration after the application of the full sweeping preconditioner acting on the Schwarz global system.

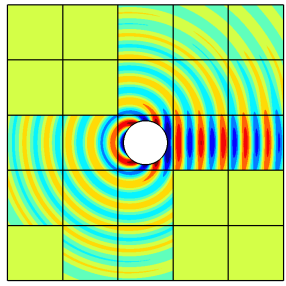
Figures 4.1c-4.1d and 4.1e-4.1f show the same information at iterations 2 and 3, respectively.

4.3 Sweeping preconditioners with one cut

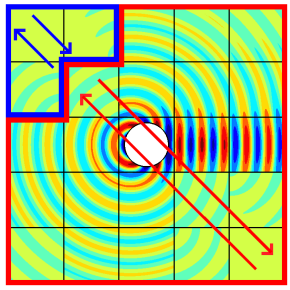
Here, the idea is to introduce a cut on one of the transmission boundaries in the sweeping domain (Figure 4.2). In Figure 4.2, one square stands for one



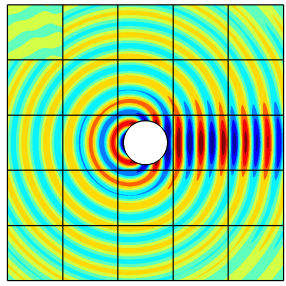
(A) Iteration 1, the application of $\mathcal{F}$



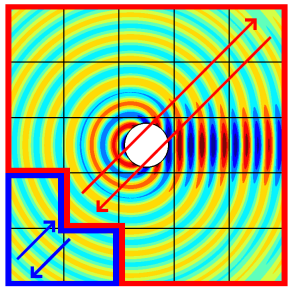
(B) Iteration 1, the application of $\mathcal{F}^{-1}$



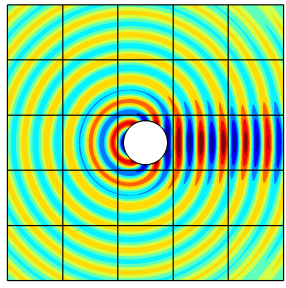
(C) Iteration 2, the application of $\mathcal{F}$



(D) Iteration 2, the application of $\mathcal{F}^{-1}$



(E) Iteration 3, the application of $\mathcal{F}$



(F) Iteration 3, the application of $\mathcal{F}^{-1}$

FIGURE 4.3: Visualisation: sweeping preconditioners with one cut.

diagonal group in a checkerboard domain decomposition. Therefore, there are 6 groups in Figure 4.2.

The preconditioner with one cut can be derived from the full Schwarz preconditioner matrix. In Equation (4.1), the preconditioner for the left-top box is similar to the full preconditioner with two diagonal groups. Thus, the preconditioner with one cut can be considered as combining two full preconditioners. The difference is that identity operators are "overlap" because the interface between the second group and the third group is shared by two boxes. For the sweeping domain in (4.1), the left-top box consists of two groups and

the right-bottom box has $s-2$ groups:

$$\begin{bmatrix} \mathcal{I}^{[1]}_{[1]} & \mathcal{F}^{[2]}_{[1]} & & & & & \\ \mathcal{F}^{[1]}_{[2]} & \mathcal{I}^{[2]}_{[2]} & \mathcal{F}^{[3]}_{[2]} & & & & \\ & \mathcal{F}^{[2]}_{[3]} & \mathcal{I}^{[3]}_{[3]} & \mathcal{F}^{[4]}_{[3]} & & & \\ & & \mathcal{F}^{[3]}_{[4]} & \mathcal{I}^{[4]}_{[4]} & \ddots & & \\ & & & \ddots & \ddots & \mathcal{F}^{[s]}_{[s-1]} & \\ & & & & \mathcal{F}^{[s-1]}_{[s]} & \mathcal{I}^{[s]}_{[s]} & \end{bmatrix} \begin{bmatrix} \mathbf{g}_{[1]} \\ \mathbf{g}_{[2]} \\ \mathbf{g}_{[3]} \\ \vdots \\ \mathbf{g}_{[s-1]} \\ \mathbf{g}_{[s]} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_{[1]} \\ \mathbf{b}_{[2]} \\ \mathbf{b}_{[3]} \\ \vdots \\ \mathbf{b}_{[s-1]} \\ \mathbf{b}_{[s]} \end{bmatrix}, \tag{4.1}$$

Visually, in Figure 4.3:

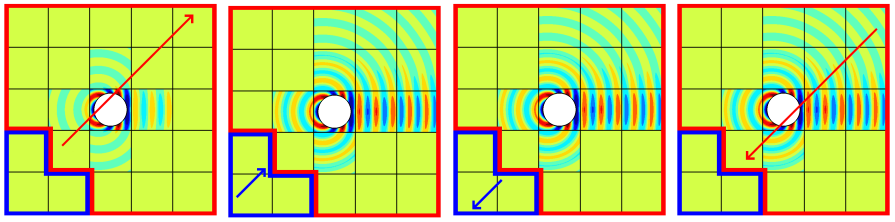
- Figure 4.3a shows the wave propagation at the 1st iteration of the iterative solution of the global Schwarz system, with red and blue arrows indicating coming application, i.e. preconditioning with one cut.
- Figure 4.3b shows the wave propagation at the 1st iteration after the application of the sweeping preconditioner with one cut acting on Schwarz global system.

Figures 4.3c-4.3d and 4.3e-4.3f show the same information at iterations 2 and 3, respectively.

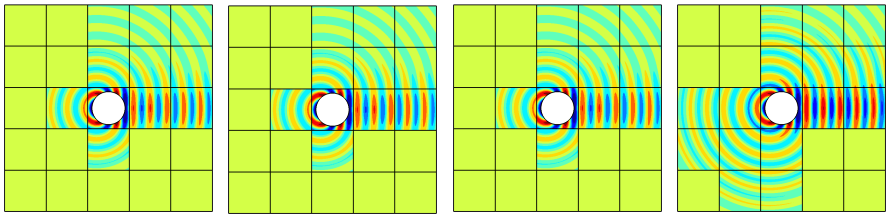
4.4 Parallelization strategy

Let us recall that the transmission matrix $\mathcal{F}^{[j]}_{[i]}$ operating on variable $\mathbf{g}_{[j]}$ describes the information transferred from j^{th} group to i^{th} group. So the matrix operating on the variable $\mathcal{F}^{[j]}_{[i]}\mathbf{g}_{[j]}$ $(i > j)$ is the j^{th} *forward step* and $\mathcal{F}^{[j]}_{[i]}\mathbf{g}_{[j]}$ $(i < j)$ is the j^{th} *backward step*. The group number is equal to the step number which the group occupies. Let us sort the group numbers from bottom-left to top-right, with number $1, 2, 3, \ldots, s$.

We can now present a parallelization strategy for multiple sources, which combines both the full sweeping preconditioner and the sweeping preconditioner with one cut, with the help of Figures 4.3a to 4.3b. Figure 4.4a shows the coming sweeping preconditioning (3^{rd} forward step to 9^{th} forward step) in the red region and the result after this preconditioning is shown in Figure 4.4e. Then, Figure 4.4b shows the coming sweeping preconditioning (1^{st} forward step to 2^{nd} forward step) in the blue region and the result after this



(A) before 3^{rd} forward step to 9^{th} forward step (B) before 1^{st} forward forward step to 2^{nd} forward step (C) before 2^{nd} backward step to 1^{st} backward step (D) before 9^{th} backward step to 3^{rd} backward step



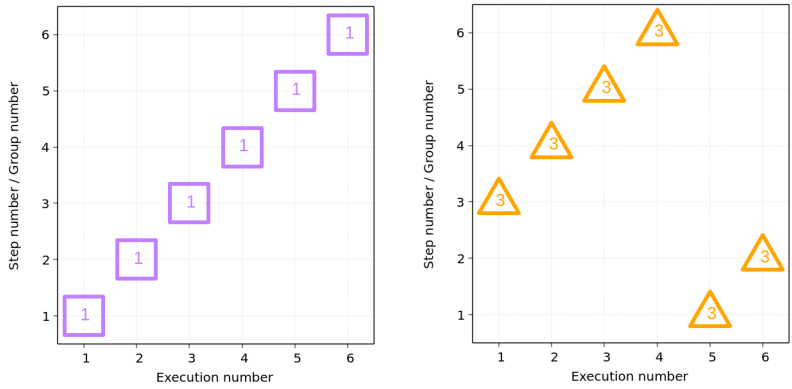
(E) after 3^{rd} forward step to 9^{th} forward step (F) after 1^{st} forward forward step to 2^{nd} forward step (G) after 2^{nd} backward step to 1^{st} backward step (H) after 9^{th} backward step to 3^{rd} backward step

FIGURE 4.4: Visualisation: detail in iteration 1 of the sweeping preconditioner with one cut.

preconditioning is shown in Figure 4.4f. We can see that each group can occupy one forward step or backward step and these groups can complete one sweeping preconditioning with one cut. For the sake of clarity, we put these groups and steps in a graph (see Figure 4.5b in which there are 6 diagonal groups and 6 steps). In addition, Figure 4.5a shows how the groups complete one full sweeping preconditioning sequentially. Finally, we can see that the difference between the strategy of full sweeping preconditioning and that of sweeping preconditioning with one cut is the starting group.

It is obvious that Figure 4.5a and Figure 4.5b can merge into one graph which means these groups can complete sweeping preconditioning for different sources in parallel (see Figure 4.6a). At the first execution, the first group performs the first forward step for source 1 and the third group performs the third forward step for source 3. After all executions, these groups complete sweeping preconditioning for source 1 and source 2 simultaneously. Furthermore, there is no difficulty in completing sweeping preconditioning for 6 sources simultaneously (see Figure 4.5b).

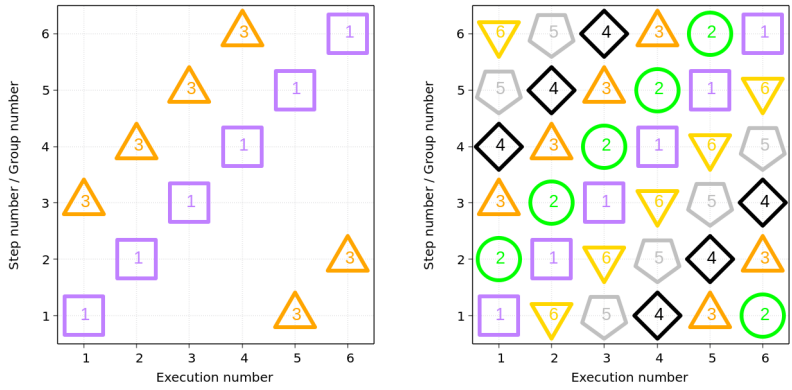
Until now, we have described the parallelization strategy by combining full sweeping preconditioner and sweeping preconditioner with one cut. Each group occupies one forward step or backward step and they complete one sweeping preconditioning sequentially. For subdomains in one group, each



(A) Full preconditioning without cut for 3^{th} source.

(B) Preconditioning with one cut for 3^{th} source.

FIGURE 4.5: Execution: for one same source, each group occupies one forward step and backward step and these groups complete one sweeping preconditioning sequentially.



(A) Preconditionings for 1^{st} source & 3^{rd} source.

(B) Preconditionings for 1^{st}, 2^{nd}, …, 6^{th} sources.

FIGURE 4.6: Parallelization strategy: combination of sweeping preconditionings for multiple sources. Numbers at the center of different geometric shapes represent the source numbers.

subdomain is assigned to one thread. In the other words, we choose the one-thread-to-one-subdomain fashion to realize this parallel strategy if the number of sources is equal to the number of groups.

Considering that the number of sources is much less than the number of

groups, we can choose a nested parallel strategy for improving the usage of computer resources. For instance, if there are $2N$ groups and N sources, we can put two threads in each subdomain.

4.5 Time complexity

For two-dimensional Helmholtz simulations without DD methods, after the factorization phase, a solution of its sparse matrical system is computed in $O(n)$ run-time in a sequential computational environment, where n is the total number of degrees of freedom. Let us consider DD methods and assume that there are $r \times (t \cdot r)$ subdomains, where r and $t \cdot r$ are the numbers of rows and columns of non-overlapping subdomains ($N_{\text{dom}} = r \times (t \cdot r)$), and that there are tr^2 threads, i.e. that each thread is assigned to one subdomain.

First, let's discuss the time complexity of the (flexible) GMRES (the factorization has been done and the commnunication complexity and basic vector operations in the Arnoldi process are ignored) of optimized Schwarz methods without sweeping preconditioners and with a single RHS. The complexity per thread is

$$O\left(\frac{n}{tr^2}\right) \times O(r + tr - 1) = O\left(\frac{n}{r}\right), \tag{4.2}$$

where the left side corresponds to the application of $\mathcal{F}$. Its computation in each subdomain can be done in parallel, which leads to a complexity proportional to the total number of degrees-of-freedom divided by the number of threads. For the homogeneous cases, the number of iterations is empirically equal to the number of groups of subdomains, which is $O(r + tr - 1)$ (for waveguide problems, this statement is only valid for the layered domain decompositions, that is $r = 1$).

Next, we discuss the time complexity per thread with sweeping preconditioners:

$$\left(O\left(\frac{n}{tr^2}\right) + O\left(\frac{n}{tr^2}\right) \times (r + tr - 1)\right) \times O(1) = O\left(\frac{n}{r}\right),$$

which includes the application of $\mathcal{F}$ and of $\mathcal{F}^{-1}$, i.e. the two main parts in each iteration of (F)GMRES. We have shown that the application of $\mathcal{F}^{-1}$ contains forward steps and backward steps in one iteration of (F)GMRES. Forward steps (or backward steps) are equal to the number of groups, and they must be performed sequentially (they can only be executed step by step). Therefore, the application of $\mathcal{F}^{-1}$ leads to a complexity of $(r + tr - 1)$ times $O(n/(tr^2))$. Thanks to the effectiveness of our generalized preconditioners, the number of iterations of (F)GMRES is reduced to $O(1)$.

Third, we discuss the time complexity with sweeping preconditioners and with as many RHS as diagonal groups $(r + tr - 1)$. The time complexity per

thread of the (flexible) GMRES is:

$$\left(O\left(\frac{n\times(r+tr-1)}{tr^2}\right)+O\left(\frac{n\times(r+tr-1)}{tr^2}\right)\times(r+tr-1)\right)\times O(1)=O\left(tn\right).$$

Finally, let's discuss the case with the improved parallelization strategies with as many RHS as there are diagonal groups in a checkerboard partition. Figure 4.6b shows that every thread works at every sequential step and there is no waiting. Instead of solving all RHS together, the parallelization strategy allow one thread to only compute one RHS at each step and, however, to calculate all RHS simultaneously. Thus, the complexity per thread of the (flexible) GMRES of the same cases is

$$\left(O\left(\frac{n\times(r+tr-1)}{tr^2}\right)+O\left(\frac{n}{tr^2}\right)\times(r+tr-1)\right)\times O(1)=O\left(\frac{n}{r}\right).$$

From this complexity analysis, it is worth noting that the speedup potential of these improved parallelization strategies for many right-hand sides can be boosted further by increasing the number of subdomains.

4.6 Numerical results

In this section, the improved parallelization strategies are compared and studied by using two two-dimensional benchmarks. We consider both the runtime of (F)GMRES in a waveguide problem with layered domain decompositions, and the number of iterations of a scattering problem in free space with checkerboard domain decompositions. By studying the convergence rate of the improved parallelization strategies in the latter case, we can check if these strategies maintain the effectiveness of our generalized preconditioners.

4.6.1 Waveguide model

First, we apply our improved parallelization strategies for waveguide problems. We consider the same homogeneous waveguide problem as in 2.4.2, and analyze the performance of the optimized Schwarz DD method with layered decompositions preconditioned by the double sweep preconditioner. The resulting discrete system is solved with as many right-hand sides as the number of subdomains. The theoretical complexity per thread of the (flexible) GMRES without preconditioner is $(r+tr-1)$ times equation (4.2), which leads to a theoretical complexity of $O(tn)$. Therefore, the ratio of the time complexity improved by parallelization strategies and the time complexity without preconditioners is

$$O\left(\frac{n}{r}\right)/O(tn)=O\left(\frac{1}{tr}\right).$$

	$\omega = 20\pi$				
	$N_{\text{dom}} = 5$	10	25	50	100
Improved parallel.	$3s$	$3s$	$5s$	$11s$	$21s$
Unpreconditioned	$2s$	$5s$	$14s$	$42s$	$319s$
Ratio	1.50	0.60	0.36	0.26	0.07

TABLE 4.1: Homogeneous waveguide with $1 \times t = 1 \times N_{\text{dom}}$ layered domain decomposition: runtime of GMRES as a function of N_{dom} to reach a relative residual of 10^{-6}. The number of RHS is equal to N_{dom}.

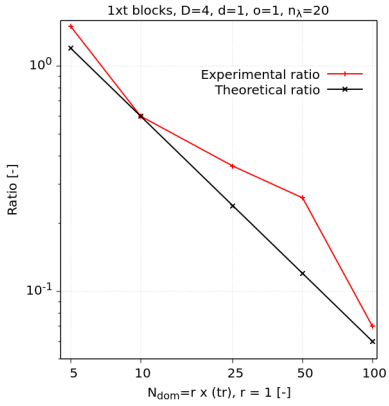


FIGURE 4.7: Homogeneous waveguide with $1 \times t$ layered domain decompositions: ratio between the computing times obtained with the improved parallelization strategies and the unpreconditioned solver ("Experimental ratio") and the theoretical scaling $\frac{6}{tr}$ ("Theoretical ratio").

In the case of layered decompositions, the number of rows r is equal to 1. Table 4.1 reports the GMRES runtime with the improved parallelization strategies and that with the unpreconditioned algorithm, as well as the measured ratio. The proposed improvements are clearly very effective to accelerate the solution, especially with a large number of subdomains (and right-hand sides). As shown in Figure 4.7, the measured computing time ratio nicely reflects the theoretical scaling, which proves the effectiveness of the proposed improved parallelization strategies.

4.6.2 Scattering model in free space

Finally, we apply our improved parallelization strategies for scattering problems in free space. We consider the same homogeneous scattering problem

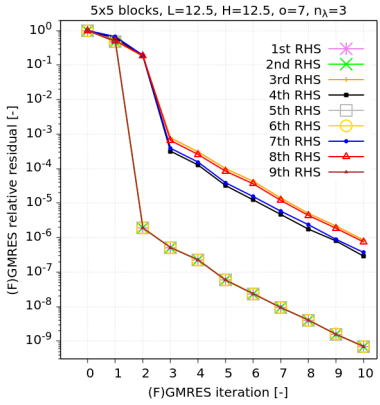


FIGURE 4.8: Scattering problem with multiple sources. Residual history with P7 triangles and 3 elements by wavelength.

as in Section 3.3.1 (second configuration, with the source in the middle of the domain). We analyze the history of convergence of optimized Schwarz methods with block decompositions preconditioned by the double sweep preconditioner.

We consider 9 right-hand sides, which is equal to the number of groups. From Figure 4.8, the histories of 1^{st}, 2^{nd}, 5^{th}, 6^{th} and 9^{th} RHS are observed to be indentical. Indeed, for the 1^{st}, 2^{nd} and 9^{th} RHS, sweeping preconditioners are equal to full sweeping preconditioners, and for the 5^{th} and 6^{th} RHS, sweeping preconditioners are cut at the group where the source is placed. Hence these two preconditioners are also equivalent to full sweeping preconditioners. The other sweeping preconditioners for the 3^{rd}, 4^{th}, 7^{th} and 8^{th} RHS are sweeping preconditioners with one cut, which can be illustrated in Figure 4.3. It is expected that the convergence rate of these sweeping preconditioners with one cut is slower, which determines the finalization of the simulation for all right-hand sides. The convergence rate of the improved parallelization strategies is thus seen to not deteriorate too much, and its effectiveness can thus be expected to be close to the theoretical prediction provided above as well.

4.7 Conclusion

In this chapter we have shown that the full sweeping preconditioner and the sweeping preconditioner with one cut can be combined to accelerate the solution of Helmholtz problems with multiple right-hand sides. Instead of waiting for other threads, one thread works in every step of one sweeping preconditioner, which maximizes the usage of computers' resources and maintains the fast rate of convergence. These improved parallelization strategies will

undoubtedly be useful for imaging applications where the same Helmholtz equation has to be solved with many sources, and the same principle should be applicable as-is to electromagnetic and elastodynamic problems. Future work should investigate the communication complexity and heterogeneous cases, both of which are non-trivial for extremely large problems.

General conclusion and perspectives

A research journey

This work has its origin in a well-known problem in the field of scientific computing. Solving the Helmholtz equation in the frequency domain using finite elements leads to indefinite complex-valued matrices that are hard to invert. This difficulty is even more important in the case of high frequency: the linear systems are all the larger and the undefined character is all the more pronounced as the frequency increases.

The iterative methods that are widely used in CFD have so far not shown their efficiency in high frequency problems. Direct solvers have been the only option for obtaining solutions. Nevertheless, in dimension three, solving a problem with n unknowns using a direct solver requires a storage of $O(n^{4/3})$ complex numbers and the number of operations required for solving a linear system scales like $O(n^2)$. This limits the use of direct methods to problems of size say $n \simeq 10^6$ on one single core. It is indeed possible to parallelize direct methods but the n^2 FLOPS limitation prohibits their use for very large problems: if one core can be used for $n = 10^6$, we may need over 10^6 cores for $n = 10^9$!

Therefore, the scientific computing community has been looking for alternative approaches and the most promising one is based on the idea of *domain decomposition* (DD). The idea of a DD method is to decompose the domain of interest spatially into subdomains that can be solved individually using a direct solver. The main research question in DD methods is to find the optimal way to exchange information between subdomains with the aim of getting all subdomains to *agree* and thus converge to the solution on the whole domain.

Even though naive DD methods converge to the solution, the number of iterations of such naive DD methods grows with the number of subdomains. It is possible to use preconditionners that allow to converge in a number of iterations that is quasi independent of the number of subdomains. Disappointingly, such preconditionners are intrinsically sequential which limits their efficiency. This was the starting point of this Ph.D. work.

The first part of this work has been dedicated to the efficient implementation of a high-order finite element Helmholtz solver. Starting from the adage

that says that a chain is never better than the worst of its links, we took a lot of time to make that first building block as efficient as possible. At first, we have re-designed the standard element-by-element finite element assembly procedure as a computationally efficient large matrix-matrix multiplication. The use of BLAS subroutines has allowed us to reach assembly performances that are close to the peek performance of computers. We then used the most efficient linear system solver available to obtain solutions. At that point, we were confident to start to work on the core of our work, namely the DD solver.

Optimized Schwarz methods are considered in this dissertation. This approach actually couples direct and iterative strategies.

Firstly, we implemented optimized Schwarz methods with layered domain decomposition configurations in order to avoid to cross-points where more than two subdomains meet. The convergence rate of our first attempt strongly depends on the number of subdomains. This is essentially due to the layered-type domain decompositions and its sequential nature. This severe drawback forbids the use of layered subdomains for large-scale applications. We thus decided to use optimized Schwarz methods on more flexible block-type domain decompositions which might include cross-points.

Next, we succeeded in implementing optimized Schwarz methods with high-order absorbing boundary conditions and treatment of cross-points such block-type decompositions and achieved high convergence rate compared to layered-type decompositions. We must underline that even with block-type domain decompositions and with optimal transmission conditions however, the number of iterations of the domain decomposition methods will still grow as the number of subdomains increases. It is worth noting that the convergence rate without cross-points is much better than that with cross-points in terms of waveguide cases, so DD methods with layered-type domain decompositions are still valuable for waveguide problems. A nice extension of our work would be to find a way to deal with reflections and cross-points in such waveguide problems.

The next step of our work was dedicated to obtain a DD solver that is really optimal i.e. that requires a number of iterations to converge that does not depend of the number of subdomains. For that purpose, we had to find a preconditionner. We successfully developed a family of generalized sweeping preconditionners where sweeps can be performed, in parallel, in several directions, for block-type decompositions in 2D and in 3D. These sweeping preconditionners are derived in a systematic way, based on the explicit representation of the iteration matrix. With various numerical experiments, we observed that the number of iterations is not affected by the number of subdomains, and we thus obtained good parallel scalability.

Finally, we proposed a pipeline strategy for the parallel sweeping preconditionners with multiple right-hand sides considering that many real applica-

tions involve thousands of right-hand sides in their algebraic systems. This strategy combines full sweeping preconditioning procedures and cut sweeping preconditioning procedures, which maximizes the usage of computing resources and controls the number of iterations to the fullest extent.

Although these parallel preconditioned solvers are better than previous sequential preconditioned solvers, they are still not fully parallelized due to the sequential nature of sweeping. The efficiency of our parallel preconditioned solvers is not only dependent on, but also limited by processor assignments, so there is still a lot of opportunities for improvement.

Perspectives

There is a great amount of room for imagination and creativity in our research, which motivates us to continue. Here is a list of perspectives for future research and improvements to our approaches.

Research for non-homogeneous media

Although we did implement our methods for non-homogeneous media and obtained good results, it must be admitted that the Padé-type HABC that has been implemented is a *priori* suitable to wave propagation in homogeneous media. Developing such approximations for the non-homogeneous cases would improve our approaches and provide a better understanding of why current approaches work well for non-homogeneous media. While our ACE team is designing a new Padé-type HABC for non-homogeneous media, it is quite hard to deal with cross-points with this new HABC. Thus, it would be profitable to develop a new Padé-type HABC for non-homogeneous media for optimized Schwarz methods with layered-type configurations, which could be efficiently exploited when layered-type decompositions are advantageous such as in waveguide problems.

Research for optimized Schwarz methods with non-right angles

While block-type domain decompositions with right angles are quite suitable for free-space wave propagation, there are less suitable for complex geometries where defining regular domains on intricate CAD models can be very challenging. It would thus be very useful to develop optimized DD methods with cross-points with non-right angles. Recently, some new developments about non-overlapping domain decomposition methods with non-local transmission operators have been proposed [19, 21], that may serve as inspiration.

Research for multiple sweeping preconditionners

Sweeping preconditionners with layered-type configurations are sequential in nature, unless cuts are introduced. While we managed to introduce a good level of parallelism in sweeping preconditionners with block-type decompositions, each processor is still limited to occupy subdomains in one row or column. This strategy maximumized the efficiency of each processor, but however limits the number of processors. It would be interesting to investigate the use of the multi-preconditioned GMRES [47] algorithm, which allows the use of more than one preconditionner. This means that we could apply sweeping preconditionners in several directions at the same time, which could potentially further improve the parallel efficiency of our preconditioned DD methods.

Extension to eigenvalue problems

Our optimized Schwarz method shows good parallel efficiency when it treats large-scale simulations with multiple right-hand sides. When dealing with eigenvalue problems with contour integral methods [80, 9], the terms appearing in the contour integrals are nothing but a linear system with multiple right-hand sides. These could be solved with our optimized DD methods. Meanwhile, since contour integral methods can also be easily parallelized, it will be very interesting to combine these two.

Bibliography

[1] M. Ainsworth and J. T. Oden. "A posteriori error estimation in finite element analysis". In: *Computer methods in applied mechanics and engineering* 142.1-2 (1997), pp. 1–88.

[2] P. Amestoy et al. "Fast 3D frequency-domain full-waveform inversion with a parallel block low-rank multifrontal direct solver: Application to OBC data from the North Sea". In: *GEOPHYSICS* 81.6 (2016), R363–R383. DOI: 10.1190/geo2016-0052.1.

[3] X. Antoine, M. Darbas, and Y. Y. Lu. "An improved surface radiation condition for high-frequency acoustic scattering problems". In: *Computer Methods in Applied Mechanics and Engineering* 195.33 (2006). ISSN: 0045-7825.

[4] A. V. Astaneh and M. N. Guddati. "A two-level domain decomposition method with accurate interface conditions for the Helmholtz problem". In: *International Journal for Numerical Methods in Engineering* 107.1 (2016), pp. 74–90.

[5] F. V. Atkinson. "On Sommerfeld's "radiation condition."" In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 40.305 (1949), pp. 645–651.

[6] A. Bendali and Y. Boubendir. "Non-overlapping domain decomposition method for a nodal finite element method". In: *Numerische Mathematik* 103.4 (2006), pp. 515–537.

[7] A. Bermúdez et al. "An exact bounded perfectly matched layer for time-harmonic scattering problems". In: *SIAM Journal on Scientific Computing* 30.1 (2008), pp. 312–338.

[8] A. Bermúdez et al. "An optimal perfectly matched layer with unbounded absorbing function for time-harmonic acoustic scattering problems". In: *Journal of Computational Physics* 223.2 (2007), pp. 469–488. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2006.09.018.

[9] W.-J. Beyn. "An integral method for solving nonlinear eigenvalue problems". In: *Linear Algebra and its Applications* 436.10 (2012). Special Issue dedicated to Heinrich Voss's 65th birthday, pp. 3839–3863. ISSN: 0024-3795. DOI: https://doi.org/10.1016/j.laa.2011.03.030.

[10] N. Bootland et al. "A comparison of coarse spaces for Helmholtz problems in the high frequency regime". In: *arXiv preprint arXiv:2012.02678* (2020).

[11] Y. Boubendir. "An analysis of the BEM-FEM non-overlapping domain decomposition method for a scattering problem". In: *Journal of computational and applied mathematics* 204.2 (2007), pp. 282–291.

[12] Y. Boubendir, X. Antoine, and C. Geuzaine. "A Quasi-Optimal Non-Overlapping Domain Decomposition Algorithm for the Helmholtz Equation". In: *Journal of Computational Physics* 231 (Jan. 2012), pp. 262–280.

[13] Y. Boubendir and A. Bendali. "Dealing with cross-points in a non-overlapping domain decomposition solution of the Helmholtz equation". In: *Mathematical and Numerical Aspects of Wave Propagation WAVES 2003*. Springer, 2003, pp. 319–324.

[14] Y. Boubendir, A. Bendali, and M. B. Fares. "Coupling of a non-overlapping domain decomposition method for a nodal finite element method with a boundary element method". In: *International journal for numerical methods in engineering* 73.11 (2008), pp. 1624–1650.

[15] N. Bouziani, H. Calandra, and F. Nataf. *An overlapping splitting double sweep method for the Helmholtz equation*. 2021.

[16] A. Brandt and I. Livshits. "Wave-ray multigrid method for standing wave equations". In: *Electron. Trans. Numer. Anal* 6.162-181 (1997), p. 91.

[17] S. N. Chandler-Wilde and P. Monk. "Wave-Number-Explicit Bounds in Time-Harmonic Scattering". In: *SIAM Journal on Mathematical Analysis* 39.5 (2008), pp. 1428–1455. DOI: 10.1137/060662575.

[18] Z. Chen and X. Xiang. "A source transfer domain decomposition method for Helmholtz equations in unbounded domain". In: *SIAM Journal on Numerical Analysis* 51.4 (2013), pp. 2331–2356.

[19] X. Claeys. *A new variant of the Optimised Schwarz Method for arbitrary non-overlapping subdomain partitions*. 2019. arXiv: 1910.05055 [math.AP].

[20] X. Claeys. "Non-local variant of the Optimised Schwarz Method for arbitrary non-overlapping subdomain partitions". In: *ESAIM: Mathematical Modelling and Numerical Analysis* 55.2 (2021), pp. 429–448.

[21] X. Claeys and E. Parolin. *Robust treatment of cross points in Optimized Schwarz Methods*. 2020. arXiv: 2003.06657 [math.NA].

[22] P.-H. Cocquet and M. J. Gander. "How large a shift is needed in the shifted Helmholtz preconditioner for its effective inversion by multigrid?" In: *SIAM Journal on Scientific Computing* 39.2 (2017), A438–A478.

[23] F. Collino, P. Joly, and M. Lecouvez. "Exponentially convergent non overlapping domain decomposition methods for the Helmholtz equation". In: *ESAIM: Mathematical Modelling and Numerical Analysis* 54.3 (2020), pp. 775–810.

[24] L. Conen et al. "A coarse space for heterogeneous Helmholtz problems based on the Dirichlet-to-Neumann operator". In: *Journal of Computational and Applied Mathematics* 271 (2014), pp. 83–99.

[25] R. D. de Cook et al. *Concepts and applications of finite element analysis*. John wiley & sons, 2007.

[26] A. De Coninck et al. "Needles: Toward Large-Scale Genomic Prediction with Marker-by-Environment Interaction". In: 203.1 (2016), pp. 543–555.

[27] B. Després. "Méthodes de Décomposition de Domaine pour les Problèmes de Propagation d'Ondes en Régime Harmonique. Le Théorème de Borg pour l'Equation de Hill Vectorielle". PhD thesis. Paris VI University, 1991.

[28] I. S. Duff and J. K. Reid. "The Multifrontal Solution of Indefinite Sparse Symmetric Linear". In: *ACM Trans. Math. Softw.* 9.3 (Sept. 1983), pp. 302–325. ISSN: 0098-3500. DOI: 10.1145/356044.356047.

[29] H. C. Elman, O. G. Ernst, and D. P. O'Leary. "A Multigrid Method Enhanced by Krylov Subspace Iteration for Discrete Helmholtz Equations". In: *SIAM J. Sci. Comput.* 23.4 (Apr. 2001), pp. 1291–1315. ISSN: 1064-8275.

[30] B. Engquist and A. Majda. "Absorbing Boundary Conditions for the Numerical Simulation of Waves". In: *Mathematics of Computation* 31.139 (1977), pp. 629–651.

[31] B. Engquist and L. Ying. "Sweeping Preconditioner for the Helmholtz Equation: Hierarchical Matrix Representation". In: *Communications on Pure and Applied Mathematics* 64 (May 2011).

[32] B. Engquist and L. Ying. "Sweeping Preconditioner for the Helmholtz Equation: Moving Perfectly Matched Layers". In: *SIAM Journal on Multiscale Modeling and Simulation* 9 (July 2010).

[33] Y. A. Erlangga, C. V., and C. W. Oosterlee. "On a class of preconditioners for solving the Helmholtz equation". In: *Applied Numerical Mathematics* 50.3 (2004), pp. 409–425. ISSN: 0168-9274. DOI: https://doi.org/10.1016/j.apnum.2004.01.009.

[34] O. G. Ernst and M. J. Gander. "Why it is Difficult to Solve Helmholtz Problems with Classical Iterative Methods". In: *Numerical Analysis of Multiscale Problems*. Springer Berlin Heidelberg, 2012, pp. 325–363.

[35] C. Farhat and F.-X. Roux. "A method of finite element tearing and interconnecting and its parallel solution algorithm". In: *International journal for numerical methods in engineering* 32.6 (1991), pp. 1205–1227.

[36] C. Farhat et al. "A unified framework for accelerating the convergence of iterative substructuring methods with Lagrange multipliers". In: *International Journal for Numerical Methods in Engineering* 42.2 (1998), pp. 257–288.

[37] C. Farhat et al. "FETI-DPH: a dual-primal domain decomposition method for acoustic scattering". In: *Journal of Computational Acoustics* 13.03 (2005), pp. 499–524.

[38] 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?" In: *Numer. Math.* 131.3 (Nov. 2015), pp. 567–614. ISSN: 0029-599X. DOI: 10.1007/s00211-015-0700-2.

[39] M. J. Gander and F. Kwok. "On the applicability of Lions' energy estimates in the analysis of discrete optimized Schwarz methods with cross points". In: *Domain decomposition methods in science and engineering XX*. Springer, 2013, pp. 475–483.

[40] M. J. Gander, F. Magoules, and F. Nataf. "Optimized Schwarz methods without overlap for the Helmholtz equation". In: *SIAM Journal on Scientific Computing* 24.1 (2002), pp. 38–60.

[41] M. J. Gander and H. Zhang. "A class of iterative solvers for the Helmholtz equation: Factorizations, sweeping preconditioners, source transfer, single layer potentials, polarized traces, and optimized Schwarz methods". In: *SIAM Review* 61.1 (2019), pp. 3–76.

[42] R. A. van de Geijn and K. Goto. "Anatomy of high-performance matrix multiplication Kazushige Goto, Robert A. van de Geijn ACM Transactions on Mathematical Software (TOMS), 2008". In: *ACM Transactions on Mathematical Software* 34 (May 2008), Article 12.

[43] C. Geuzaine and J.-F. Remacle. "Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities". In: *International journal for numerical methods in engineering* 79.11 (2009), pp. 1309–1331.

[44] D. Givoli. "Computational Absorbing Boundaries". In: *Computational Acoustics of Noise Propagation in Fluids - Finite and Boundary Element Methods*. Ed. by S. Marburg and B. Nolte. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 145–166. ISBN: 978-3-540-77448-8. DOI: 10.1007/978-3-540-77448-8_6.

[45] D. Givoli. "High-order local non-reflecting boundary conditions: a review". In: *Wave Motion* 39.4 (2004). New computational methods for wave propagation, pp. 319–326.

[46] D. Givoli. *Numerical Methods for Problems in Infinite Domains*. Advances in Psychology. Elsevier, 1992. ISBN: 9780444888204.

[47] C. Greif, T. Rees, and D. B. Szyld. "GMRES with multiple preconditioners". In: *SeMA Journal* 74.2 (2017), pp. 213–231.

[48] J. A. Gunnels, G. M. Henry, and R. A. van de Geijn. "A Family of High-Performance Matrix Multiplication Algorithms". In: *Applied Parallel Computing. State of the Art in Scientific Computing*. Springer Berlin Heidelberg, 2006, pp. 256–265.

[49] A. Heinecke et al. "Design and Implementation of the Linpack Benchmark for Single and Multi-node Systems Based on Intel® Xeon Phi Coprocessor". In: *2013 IEEE 27th International Symposium on Parallel and Distributed Processing*. 2013, pp. 126–137.

[50] K. Hillewaert. "Development of the discontinuous Galerkin method for high-resolution, large scale CFD and acoustics in industrial geometries". In: 2013.

[51] K. Hillewaert. "Exploiting Data Locality in the DGM Discretisation for Optimal Efficiency". In: *ADIGMA - A European Initiative on the Development of Adaptive Higher-Order Variational Methods for Aerospace Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 11–23.

[52] F. Ihlenburg and I. Babuska. "Finite Element Solution of the Helmholtz Equation with High Wave Number Part II: The h-p Version of the FEM". In: *SIAM Journal on Numerical Analysis* 34.1 (1997), pp. 315–358.

[53] J. Jeffers, J. Reinders, and A. Sodani. *Intel Xeon Phi Processor High Performance Programming: Knights Landing Edition*. Elsevier Science, 2016.

[54] P. Joly, S. Lohrengel, and O. Vacus. "Un résultat d'existence et d'unicité pour l'équation de Helmholtz avec conditions aux limites absorbantes d'ordre". In: *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics* 329.3 (1999), pp. 193–198.

[55] M. Kaltenbacher. *Computational Acoustics*. Springer, 2018.

[56] R. Kechroud, X. Antoine, and A. Soulaimani. "Numerical accuracy of a Padé-type non-reflecting boundary condition for the finite element solution of acoustic scattering problems at high-frequency". In: *International Journal for Numerical Methods in Engineering* 64.10 (2005), pp. 1275–1302.

[57] D. Kourounis, A. Fuchs, and O. Schenk. "Towards the Next Generation of Multiperiod Optimal Power Flow Solvers". In: *IEEE Transactions on Power Systems* PP.99 (2018), pp. 1–10.

[58] A. de La Bourdonnaye et al. "A Non-Overlapping Domain Decomposition Method for the Exterior Helmholtz Problem". In: *Contemporary Mathematics* 218 (1998), pp. 42–66.

[59] D. Lahaye, J. Tang, and K. Vuik. *Modern Solvers for Helmholtz Problems*. Springer International Publishing, 2017.

[60] A. Laird and M. Giles. "Preconditioned Iterative Solution of the 2D Helmholtz Equation". In: (Jan. 2002).

[61] J. Lambrechts. "Finite element methods for coastal flows: application to the Great Barrier Reef". PhD thesis. UCL-Université Catholique de Louvain, 2011.

[62] M. Lecouvez et al. "Quasi-local transmission conditions for non-overlapping domain decomposition methods for the Helmholtz equation". In: *Comptes Rendus Physique* 15.5 (2014), pp. 403–414.

[63] W. Leng and L. Ju. *A diagonal sweeping domain decomposition method with source transfer for the Helmholtz equation*. Preprint arXiv 2002.05327. 2020. arXiv: 2002.05327.

[64] R. Lim et al. "An implementation of matrix–matrix multiplication on the Intel KNL processor with AVX-512". In: *Cluster Computing* 21 (Dec. 2018).

[65] E. L. Lindman. ""Free-space" boundary conditions for the time-dependent wave equation". In: *Journal of Computational Physics* 18.1 (1975), pp. 66–78.

[66] P.-L. Lions. "On the Schwarz alternating method III: a variant for nonoverlapping subdomains". In: *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations*. 1990.

[67] P.-L. Lions. "On the Schwarz alternating method. I". In: *First international symposium on domain decomposition methods for partial differential equations*. Vol. 1. Paris, France. 1988, p. 42.

[68] N. Marsic. "Efficient methods for large-scale time-harmonic wave simulations". In: 2016.

[69] F. A. Milinazzo, C. A. Zala, and G. H. Brooke. "Rational square-root approximations for parabolic equation algorithms". In: *The Journal of the Acoustical Society of America* 101.2 (1997), pp. 760–766.

[70] A. Modave, C. Geuzaine, and X. Antoine. "Corner treatments for high-order local absorbing boundary conditions in high-frequency acoustic scattering". In: *Journal of Computational Physics* 401 (2020), p. 109029. ISSN: 0021-9991.

[71] A. Modave et al. "A non-overlapping domain decomposition method with high-order transmission conditions and cross-point treatment for Helmholtz problems". In: *Computer Methods in Applied Mechanics and Engineering* 368 (2020), p. 113162.

[72] A. Moiola and E. A. Spence. "Is the Helmholtz Equation Really Sign-Indefinite?" In: *SIAM Review* 56.2 (2014), pp. 274–312. DOI: 10.1137/120901301.

[73] F. Nataf. *On the use of open boundary conditions in block Gauss-Seidel methods for the convection-diffusion equations*. Tech. rep. CMAP (Ecole Polytechnique), 1993.

[74] F. Nataf and F. Nier. "Convergence rate of some domain decomposition methods for overlapping and nonoverlapping subdomains". In: *Numerische mathematik* 75.3 (1997), pp. 357–377.

[75] F. Nataf, F. Rogier, and E. de Sturler. "Optimal interface conditions for domain decomposition methods". In: *CMAP (Ecole Polytechnique)* 301 (1994), pp. 1–18.

[76] A. Nicolopoulos. "Formulations variationnelles d'équations de Maxwell résonantes et problèmes aux coins en propagation d'ondes". PhD thesis. Sorbonne Université, 2019.

[77] D. Rabinovich, D. Givoli, and E. Bécache. "Comparison of high-order absorbing boundary conditions and perfectly matched layers in the frequency domain". In: *International Journal for Numerical Methods in Biomedical Engineering* 26.10 (2010), pp. 1351–1369.

[78] Y. Saad. "A Flexible Inner-Outer Preconditioned GMRES Algorithm". In: *SIAM Journal on Scientific Computing* 14.2 (1993), pp. 461–469.

[79] Y. Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.

[80] T. Sakurai and H. Sugiura. "A projection method for generalized eigenvalue problems using numerical integration". In: *Journal of Computational and Applied Mathematics* 159.1 (2003). 6th Japan-China Joint Seminar on Numerical Mathematics; In Search for the Frontier of Computational and Applied Mathematics toward the 21st Century, pp. 119–128. ISSN: 0377-0427. DOI: https://doi.org/10.1016/S0377-0427(03)00565-X.

[81] H. A. Schwarz. *Gesammelte mathematische Abhandlungen*. volume 2. Springer, Berlin, 1890, pp. 133–143.

[82] H. A. Schwarz. *Ueber einen Grenzübergang durch alternirendes Verfahren.* Zürcher u. Furrer, 1870.

[83] P. Solin, K. Segeth, and I. Dolezel. *Higher-Order Finite Element Methods.* Studies in Advanced Mathematics. CRC Press, 2003.

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

[85] C. C. Stolk. "An improved sweeping domain decomposition preconditioner for the Helmholtz equation". In: *Advances in Computational Mathematics* 43.1 (2017), pp. 45–76.

[86] J. W. Strutt. *The Theory of Sound.* Vol. 2. Cambridge Library Collection - Physical Sciences. Cambridge University Press, 2011. DOI: 10.1017/CBO9781139058094.

[87] M. Taus et al. "L-Sweeps: A scalable, parallel preconditioner for the high-frequency Helmholtz equation". In: *arXiv preprint arXiv:1909.01467* (2019).

[88] A. Toselli and O. Widlund. *Domain decomposition methods-algorithms and theory.* Vol. 34. Springer Science & Business Media, 2006.

[89] F. Verbosio et al. "Enhancing the scalability of selected inversion factorization algorithms in genomic prediction". In: *Journal of Computational Science* 22.Supplement C (2017), pp. 99–108.

[90] A. Vion. "Multi-domain approaches for the solution of high-frequency time-harmonic propagation problems". PhD thesis. Université de Liège, Liège, Belgique, 2014.

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

[92] A. Vion and C. Geuzaine. "Improved sweeping preconditioners for domain decomposition algorithms applied to time-harmonic Helmholtz and Maxwell problems". In: *ESAIM: Proceedings and Surveys* 61 (2018), pp. 93–111.

[93] A. Vion et al. "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).

[94] S. Wang et al. “Massively parallel structured multifrontal solver for time-harmonic elastic waves in 3-D anisotropic media”. In: *Geophysical Journal International* 191.1 (Oct. 2012), pp. 346–366. ISSN: 0956-540X. DOI: 10.1111/j.1365-246X.2012.05634.x. eprint: https://academic.oup.com/gji/article-pdf/191/1/346/5896715/191-1-346.pdf.

[95] Z. Xianyi, W. Qian, and Z. Yunquan. “Model-driven Level 3 BLAS Performance Optimization on Loongson 3A Processor”. In: *IEEE 18th International Conference on Parallel and Distributed Systems*. 2012, pp. 684–691.

[96] L. Zepeda-Núñez and L. Demanet. “The method of polarized traces for the 2D Helmholtz equation”. In: *Journal of Computational Physics* 308 (2016), pp. 347–388.