ABSTRACT

EFFICIENT DOMAIN DECOMPOSITION ALGORITHMS FOR THE SOLUTION OF THE HELMHOLTZ EQUATION

by
Dawid Midura

The purpose of this thesis is to formulate and investigate new iterative methods for the solution of scattering problems based on the domain decomposition approach. This work is divided into three parts. In the first part, a new domain decomposition method for the perfectly matched layer system of equations is presented. Analysis of a simple model problem shows that the convergence of the new algorithm is guaranteed provided that a non-local, square-root transmission operator is used. For efficiency, in practical simulations such operators need to be localized. Current, state of the art domain decomposition algorithms use the localization technique based on rational approximation of the symbol of the transmission operator. However, the original formulation of the procedure assumed decompositions that contain no cross-points and consequently could not be used in the cross-point algorithm. In the context of the perfectly matched layer problem, we adapt the cross-point technique and combined with the rational approximation of the square root transmission operator to yield an effective algorithm. Furthermore, to reduce Krylov subspace iterations, we present a new, adequate and efficient preconditioner for the perfectly matched layer problem. The new, zero frequency limit preconditioner shows great reduction in the required number of iterations while being extremely easy to construct.

In the second part of the thesis, a new domain decomposition algorithm is considered. From theoretical point of view, its formulation guarantees well-posedness of local problems. Its practicality on the other hand is evident from its efficiency and ease of implementations as compared with other, state of the art domain decomposition approaches. Moreover, the method exhibits robustness with respect to the problem frequency and is suitable for large scale simulations on a parallel computer.

Finally, the third part of the thesis presents an extensible, object oriented architecture that supports development of parallel domain decomposition algorithms where local problems are solved by the finite element method. The design hides mesh implementation details and is capable of supporting various families of finite elements together with quadrature formulas of suitable degree of precision.

EFFICIENT DOMAIN DECOMPOSITION ALGORITHMS FOR THE SOLUTION OF THE HELMHOLTZ EQUATION

by
Dawid Midura

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology and
Rutgers, The State University of New Jersey – Newark
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Mathematical Sciences

Department of Mathematical Sciences, NJIT
Department of Mathematics and Computer Science, Rutgers-Newark

August 2014

APPROVAL PAGE

EFFICIENT DOMAIN DECOMPOSITION ALGORITHMS FOR THE SOLUTION OF THE HELMHOLTZ EQUATION

Dawid Midura

BIOGRAPHICAL SKETCH

Author:	Dawid Midura
Degree:	Doctor of Philosophy
Date:	August 2014

Undergraduate and Graduate Education:

- Doctor of Philosophy in Mathematical Sciences,
  New Jersey Institute of Technology, Newark, NJ, 2014
- Bachelor of Science in Computer Science,
  New Jersey Institute of Technology, Newark, NJ, 2009
- Bachelor of Science in Mathematical Sciences,
  New Jersey Institute of Technology, Newark, NJ, 2009

Major:	Mathematical Sciences

Presentations and Publications:

I dedicate my dissertation work to my family and friends who never failed to encourage and support me. A special feeling of gratitude to my loving parents Maria and Arkadiusz Midura without whom I would not be where I am now.

ACKNOWLEDGMENT

I would like to thank my thesis adviser, Dr. Yassine Boubendir for his continued support, and a sense of direction which he never failed to provide. I would like to give special thanks to Dr. Peter Petropoulos for sharing his ideas and providing valuable insight into our work. I also would like to thank the whole community at the Mathematical Sciences Department at NJIT. I would not have been at this point in my life without their support.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF TABLES

LIST OF TABLES
(Continued)

Table **Page**

LIST OF FIGURES

Figure **Page**

LIST OF FIGURES (Continued)

CHAPTER 1

SCATTERING PROBLEM

1.1 Problem Motivation

This thesis develops efficient computational methods for the solution of the acoustic scattering problem. Our aim is to model the dynamics of propagation of a time-harmonic wave scattered through a homogeneous medium by an obstacle of finite volume. The obstacle may be penetrable, in which case it may be filled with an inhomogenous material. The obstacle is represented as a fixed compact subset of the Euclidean space $\mathbb{R}^d$ where $d = 1, 2$ or 3. Our goal is an efficient determination of the amplitude of the scattered wave which corresponds to the radiating solution of the exterior Helmholtz boundary value problem. We should point out that while the model addresses simulation of acoustic waves, electromagnetic and elastic waves are modeled using similar mathematical principles. In consequence, numerical investigations into the wave propagation phenomena in respective fields of electromagnetism or elasticity may benefit from our work as well.

Efficient numerical modeling of wave scattering phenomena is at the heart of a number of scientific and technological fields. A direct scattering problem is concerned with determination of the wave pattern given the incident field and the shape of the obstacle. Solution to this problem finds many applications in the civilian and military sectors. For instance, the design of aircraft or submarines with respect to their stealth properties relies on an efficient evaluation of the wave field around those objects. An efficient solution to the forward problem is indispensable in these cases. Medical and seismic imaging is also an important area of application where the inverse problem is mainly of interest. Ultrasound technology and fossil fuel explorations are based on a computer generated solution to the inverse scattering problem. An inverse scattering problem deals with determination of the obstacle's shape and properties under the assumption of knowledge of both, the incident and scattered fields. It turns out that the ability to efficiently solve the direct problem addressed in this thesis is also required for an efficient solution to the inverse problem which further compounds its applicability. Waveguide and antennae design are additional areas that could benefit from the work presented here.

1.2 Derivation of the Helmholtz Equation

Mathematical acoustics is concerned with modeling of sound waves generated as a result of small perturbations of an inviscid, compressible fluid. The equations of acoustics are obtained by linearization of the equations of motion describing such a fluid about a quiescent state. More precisely, let $\mathbf{x} \in \mathbb{R}^d$ where $d = 2, 3$. We denote by p, ρ, and $\mathbf{v}$ the acoustic pressure, fluid density and fluid velocity respectively. We consider the conservation of mass and momentum equations in an inviscid fluid as the starting point. These are:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0, \tag{1.1}$$
$$\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} = -\frac{1}{\rho} \nabla p. \tag{1.2}$$

We now introduce a small disturbance to a uniform, stationary body of fluid written as:

$$p = p_0 + \tilde{p}, \qquad \rho = \tilde{\rho_0}, \qquad \mathbf{v} = \tilde{\mathbf{v}}. \tag{1.3}$$

The uniform state is $p = p_0$, $\rho(p) = \rho_0(p_0)$ and $\mathbf{v} = \mathbf{0}$ and the variables $\tilde{p}$, $\tilde{\rho}$, $\tilde{\mathbf{v}}$ are small amplitude perturbations of that state. The neglect of viscous and gravitational forces in equation (1.2) can be shown to be a valid approximation for as long as the velocity fluctuations have magnitude much less than the speed of sound and whose wavelength λ satisfies $3 \times 10^{-7}\text{m} \ll \lambda \ll 11 \times 10^{3}\text{m}$ [19]. Direct substitution into equations (1.1) and (1.2) yields:

$$\frac{\partial \tilde{\rho}}{\partial t} + \nabla \cdot \{(\rho_0 + \tilde{\rho})\, \tilde{\mathbf{v}}\} = 0, \tag{1.4}$$

$$\frac{\partial \tilde{\mathbf{v}}}{\partial t} + \tilde{\mathbf{v}} \cdot \nabla \tilde{\mathbf{v}} = -\frac{1}{\rho_0 + \tilde{\rho}} \nabla \tilde{p}. \tag{1.5}$$

Upon neglecting the products of small quantities and expanding $\frac{1}{\rho_0 + \tilde{\rho}}$ in the geometric series of $\frac{\tilde{\rho}}{\rho_0}$ we obtain the leading order equations:

$$\frac{\partial \tilde{\rho}}{\partial t} + \rho_0 \nabla \cdot \tilde{\mathbf{v}} = 0, \tag{1.6}$$

$$\frac{\partial \tilde{\mathbf{v}}}{\partial t} = -\frac{1}{\rho_0} \nabla \tilde{p}. \tag{1.7}$$

Since $\tilde{p}$ is assumed small, we may expand ρ in Taylor series about p_0:

$$\rho_0 + \tilde{\rho} = \rho(p_0 + \tilde{p}) = \rho(p_0) + \frac{d\rho}{dp}(p_0)\tilde{p} + O(\tilde{p}^2). \tag{1.8}$$

Since $\rho_0 = \rho(p_0)$ we obtain a leading order approximation:

$$\tilde{\rho} = \frac{d\rho}{dp}(p_0)\tilde{p}. \tag{1.9}$$

Deriving (1.6) with respect to time and substituting (1.9) we arrive at:

$$\nabla \cdot \frac{\partial \tilde{\mathbf{v}}}{\partial t} = -\frac{1}{\rho_0} \frac{d\rho}{dp}(p_0)\, \frac{\partial^2 \tilde{p}}{\partial t^2}. \tag{1.10}$$

Finally, we substitute (1.7) into the above and arrive at the equation satisfied by the pressure disturbance:

$$\Delta \tilde{p} = \frac{1}{c^2} \frac{\partial^2 \tilde{p}}{\partial t^2} \tag{1.11}$$

where $c = \frac{\partial \rho}{\partial p}(\rho_0)^{-\frac{1}{2}}$ is the speed of sound. This is a wave equation. Thus we see that small amplitude pressure disturbances propagate as a wave with speed c.

To decouple Equations (1.6) and (1.7), we begin by introducing function $\tilde{\phi}$ such that:

$$\frac{\partial \tilde{\phi}}{\partial t} = -\frac{\tilde{p}}{\rho_0}. \tag{1.12}$$

Equation (1.7) then implies that:

$$\frac{\partial}{\partial t}\left(\tilde{\mathbf{v}} - \nabla \tilde{\phi}\right) = 0. \tag{1.13}$$

Integrating in time and making use of the fact that $\tilde{\mathbf{v}}$ is initially zero we obtain:

$$\tilde{\mathbf{v}} = \nabla\left(\tilde{\phi} - \tilde{\phi}_0\right) \tag{1.14}$$

where $\tilde{\phi}_0$ is the initial value of $\tilde{\phi}$. Next, we introduce $\phi = \tilde{\phi} - \tilde{\phi}_0$. Substituting into the above expression yields:

$$\tilde{\mathbf{v}} = \nabla \phi. \tag{1.15}$$

In other words, ϕ is the velocity, or the acoustic potential. From the definition of ϕ and Equation (1.13), we see that:

$$\tilde{p} = -\rho_0 \frac{\partial \phi}{\partial t}. \tag{1.16}$$

Combining the above with equations (1.6), (1.9) and (1.15) gives:

$$\Delta \phi = \frac{1}{c^2}\frac{\partial^2 \phi}{\partial t^2}. \tag{1.17}$$

For time harmonic vibrations with time dependence $e^{-i\omega t}$, the velocity potential is of the form:

$$\phi(\mathbf{x}, t) = \Re\left\{u(\mathbf{x}) e^{-i\omega t}\right\} \tag{1.18}$$

with frequency $\omega > 0$. Substituting the above into (1.17) we see that the complex-valued amplitude function u satisfies the Helmholtz equation

$$\Delta u + k^2 u = 0 \tag{1.19}$$

where the constant $k = \frac{\omega}{c}$ is the wave number. We consider the problem of scattering of plane, time-harmonic acoustic waves from a bounded obstacle $\Omega \subset \mathbb{R}^d$. Therefore, we assume that the amplitude of incident acoustic potential u^i has the form $e^{i\mathbf{k}\cdot\mathbf{x}}$ where $\mathbf{k} \in \mathbb{R}^d$ with $|\mathbf{k}| = k$. The quantity $e^{i(\mathbf{k}\cdot\mathbf{x}-\omega t)}$ is constant on planes $\mathbf{k}\cdot\mathbf{x} - \omega t =$ const and so it represents a plane wave. Moreover, the wave travels in the direction $\mathbf{k}$ with speed $c = \frac{\omega}{k}$. The scattering problem is to find a scattered field u^s as the radiating solution to the Helmholtz equation in the region $\mathbb{R}^d - \Omega$ such that the total field $u = u^i + u^s$ satisfies equation (1.19) and an appropriate boundary condition on the surface of the obstacle.

The types of boundary conditions considered in acoustical applications are related to the properties of the obstacle. On the surface of the sound soft obstacle, the total excess pressure vanishes. From Equation (1.16), we see that in the time harmonic case we have the relation:

$$\tilde{p} = -i\omega\rho_0\phi \tag{1.20}$$

from which it follows that the appropriate boundary condition is of Dirichlet type:

$$u = 0 \quad \text{on} \quad \partial\Omega. \tag{1.21}$$

On the surface of a sound hard obstacle, there is no movement of fluid particles in the direction normal to the surface that is, $\mathbf{n}\cdot\tilde{\mathbf{v}} = 0$ on $\partial\Omega$. Consequently, equation (1.15) leads to a Neumann boundary condition:

$$\frac{\partial u}{\partial n} = 0 \quad \text{on} \quad \partial\Omega. \tag{1.22}$$

To characterize radiating solutions to the Helmholtz equation the *Sommerfeld radiation condition* is used

$$\lim_{r\to\infty} r^{\frac{d-1}{2}} \left(\frac{\partial u^s}{\partial r} - iku^s \right) = 0 \tag{1.23}$$

where $r = |\mathbf{x}|$ and the limit holds uniformly in all directions $\mathbf{x}/|\mathbf{x}|$. It can be shown that u^s is the part of the total field that carries the energy away from the obstacle. In other words, the Sommerfeld condition characterizes purely outgoing waves [53].

Once the solution u is known, the velocity potential is obtained by a simple multiplication by the complex exponential $e^{-i\omega t}$ and equation (1.17) immediately gives the velocity field. The associated pressure disturbance is given by equation (1.20), whereas the density disturbance can be recovered from equation (1.9).

1.3 Numerical Methods for Direct Scattering Problem

All numerical methods for modeling-time harmonic waves propagating in an unbounded medium must address the issue of an infinite problem domain. Many methods to deal with this difficulty have been proposed and studied [43, 76]. Among those methods we find absorbing boundary conditions (ABCs), the perfectly matched layer (PML) techniques, and boundary integral (BI) methods. In the case where the medium is homogeneous, the BI methods are particularly interesting since they reduce the dimensionality of the problem by one and do not require domain truncation. The other two approaches on the other hand, involve truncation of the problem domain by either an artificial surface, or an artificial absorbing layer respectively. The finite computational domain introduced in this manner encloses the scatterer, any inhomogeinities and any acoustical sources if they are present.

An artificial boundary condition closes the system by specifying the behavior of the solution at the artificial boundary. The condition needs to incorporate the far-field behavior of the solution into the numerical model. That is, it should be completely transparent to outgoing waves, and completely reflecting to incoming waves. Design of ABCs usually involves a trade off between the accuracy of computation and efficiency of the boundary condition. Local ABCs are easy evaluate and straightforward to implement in a computer code. Bayliss, Gunzburger and Turkel have show how to generate such boundary conditions by considering the far-field expansion of the solution and deriving a sequence of local operators that annihilate its first few terms [12]. Engquist and Majda [36] on the other hand used pseudo-differential operator theory to approximate the non-local Dirichlet-to-Neumann (DtN) operator by a linear combination of local operators. In the asymptotic limit, as the distance from the obstacle to the artificial boundary goes to infinity, both of these formulations reduce correctly to the Sommerfeld radiation condition. In practice of course, the distance is finite and for computational efficiency as close to the obstacle as possible. The downside to this is that these ABCs are no longer perfectly transparent and produce unwanted reflections not otherwise present in the true solution.

On the other side of the spectrum there are the non-local ABCs. They are expressed by Dirichlet to Neumann maps. The DtN operators involved are exact, and defined in terms of a convolution with the Greens function over the artificial boundary. Such boundary conditions have been discussed for example by McDonald and Wexler [60], MacCamy and Marin [59] and Givoli and Keller [51].

While local ABCs give raise to purely sparse matrices when discretized by finite elements for example, the same method generates matrices with sparse and dense blocks [32, 49] when applied to non-local ABC formulations. The sparse blocks corresponds to the application of finite element method (FEM) to the interior problem, whereas the dense blocks correspond to the application of boundary element method (BE) to the integral equation on the artificial boundary. This is problematic from the computational point of view as most linear algebra software packages are tailored for systems with either sparse or dense matrices.

To overcome the difficulties associated with the standard ABCs an alternative approach to deal with the truncation of unbounded domains was introduced by Bérenger [17] in the context of Maxwell's equations in electromagnetism. His method consisted of surrounding the computational domain with a thin layer of an artificial material designed to absorb the scattered field radiated from the obstacle. This method is referred to as "perfectly matched" because the interface between the domain of interest and the absorbing layer is transparent to the propagating waves. Though initially settled for Maxwell's equations, the PML idea has also been applied to wave problems in acoustics [46, 47], elasticity [31] and shallow water waves [63]. In practice, the thickness of the layer is finite and kept small for computational efficiency. The boundary conditions specified on the truncation boundary of the PML seem to have little effect on the method's performance [30, 64]. As compared with local ABCs, discretization of the PML system leads to larger matrices which however are still very sparse.

In the case where there are no inhomogeneities present in the problem, the boundary integral method can be used on its own [52,53]. In that case there is no need to introduce an artificial boundary since the boundary integral equation is defined directly on the surface of the scatterer. Even though the other approaches may still be employed, this technique has the advantage of being exact as the far-field behavior of the solution is completely accounted for. This method also reduces the dimensionality of the problem by one which makes it very attractive. However, we need to point out that from the discrete point of view, the associated linear system is complex, non-hermitian, dense and very large in real world problems. These properties make it very difficult to solve by direct methods. For that reason, Krylov subspace methods such as GMRES [72] for instance, are preferred instead. Large problems may further require the use of techniques such as fast multipole method (FMM) [45, 68] to decrease the computational cost of matrix-vector multiplications performed by the iterative solver. Furthermore, it should be added that integral equations arising in acoustic scattering problems can be ill-conditioned. Few equivalent integral equation formulations were thus developed to alleviate this issue on a continuous level [2,3,54,55]. Another approach is to work on a purely algebraic level, improving the conditioning of the discrete system by devising a suitable preconditioner [4,27,28].

An increasing number of applications dealing with wave scattering phenomena necessitates consideration of the problem at high frequencies. Accurate resolution of a highly oscillatory wave field by FEM or BEM requires the use of very fine meshes. This is problematic because the condition number of matrices generated by these methods grows as the mesh size decreases [6, 37]. Ill-conditioning is an undesirable property of a matrix as it makes the system difficult to solve by either direct or iterative methods. The other problem has to do with the fact that as the discretization refines, the number of mesh points increases and so does the number of the unknowns in the system. High frequency simulations using the methods presented in a direct manner are difficult to obtain simply because they give raise to systems of equations with very large number of unknowns.

The methods investigated in this thesis are based on the non-overlapping domain decomposition approach. In the domain decomposition setting, the computational domain is partitioned into a collection of non-overlapping subdomains. Small local problems coupled at the interfaces through transmission conditions are then defined on each subdomain. Subsequently, the coupling between subproblems is relaxed by formulating a suitable iterative scheme whose convergence depends on the transmission condition employed. In the process the solution to the original problem is recast as an iterative procedure in which multiple small and uncoupled problems are solved at each iteration. The advantage of this approach is that the local problem size can be kept arbitrarily small in which case the problems associated with ill-conditioning and large number of unknowns do not appear. The method has been initially conceived for elliptic type problems and much of its development can be attributed to P. L. Lions [57, 58] who considered both overlapping and non-overlapping decompositions. The extension of the non-overlapping variant of DDM to the Helmholtz problem was accomplished by B. Després [33].

Domain decomposition algorithms are very attractive from the computational point of view as they respect the memory hierarchy of modern parallel architectures. Realization of such algorithms on parallel clusters is natural due to the loose coupling between subproblems. Since local problems are small, they can be solved very efficiently on a single cluster node. Moreover, in the non-overlapping variant of the method that we are considering, only data related to the interface unknowns must be exchanged between neighboring subdomains. This means that the the communications overhead in the cluster is small which ensures robustness of such algorithms with respect to the number of subproblems.

Last, but no least, domain decomposition technique allows one to construct a coupling of algorithms between different methods of discretization, such as the FEM and the BEM, in order to solve problems involving inhomogeneous materials. FEM deals with a finite part of the domain enclosing any inhomogeinities and sources, independent of the BEM, which tackles the equations describing the propagation of the wave in an infinite homogeneous medium [16, 21, 23].

CHAPTER 2

DOMAIN DECOMPOSITION METHODS

Domain decomposition methods have a rich and interesting history [41]. The idea was first conceived by Herman Schwarz in 1869 who proposed what is now called an *alternating Schwarz method* in order to prove the validity of the Dirichlet principle on irregular domains [41]. His results were extended by P. L. Lions [57, 58] about a century later, around the time when parallel computers were becoming more available. P. L. Lions was the first to realize the potential of the Schwarz method on such machines. A slight modification to the original algorithm lead P. L. Lions to the development of the *parallel Schwarz method* [57]. His work introduced new analytical tools needed to study such algorithms and brought the Schwarz method to the modern computer era. Moreover, and what is very important, his contributions brought the fundamental ideas of decomposition and iteration, so central to the Schwarz method, to the attention of other researchers.

The basic idea of domain decomposition methods is that instead of solving one huge problem once, it may be convenient or even necessary, to solve a sequence of smaller problems iteratively [67, 74]. The original computational problem domain is first partitioned into a collection of subdomains. Subsequently, local subproblems are defined on each subdomain. Their totality composes a system of coupled boundary value problems. In the context of non-overlapping methods, the coupling between adjacent subproblems is realized via transmission conditions imposed on the artificial interfaces separating them. The choice of the transmission conditions is crucial. It defines an iteration equation whose properties directly relate to the speed of convergence of the method. The subproblems are then uncoupled in an iterative scheme. Each iteration consists of two crucial stages. In first, the local subproblems are solved independently of one another. Once the local solutions are known, the iterates are updated and exchanged in the second stage.

The ability to split a large problem into a collection of smaller ones is what makes domain decomposition algorithms very much applicable to real world problems with huge computational resource requirements. In this thesis we focus on the application of domain decomposition approach to the problem of acoustic scattering. It should be noted however, that domain decomposition techniques have successfully been applied to many large scale problems of science and engineering such as high resolution climate simulations or determination of air flow around aircraft prototypes, to mention just a few.

The Schwarz method is an overlapping domain decomposition method and one of its drawbacks is the additional amount of computational work performed in the overlapping region. Moreover, there are cases where non-overlapping decompositions are more natural to use. Such cases include problems with discontinuities where the problem domain could be partitioned along the lines of discontinuities, as well as problems involving coupling of two physical or numerical models. This has prompted P. L. Lions to study non-overlapping variants of the Schwarz method. He was able to show that the convergence of the method for Laplace type problems can still be achieved, and in fact improved, provided that a Robin type transmission condition is used instead of the Dirichlet type as was the case with the overlapping methods [58].

Unfortunately, the convergence of the non-overlapping Schwarz method devised by P. L. Lions cannot be guaranteed for all problems. Many researchers, including P. L. Lions himself were aware of this shortcoming. In [58], he pointed out that perhaps this issue can be resolved by allowing even more general Robin type boundary conditions that include non-constant coefficients or even local or non-local operators.

A well known example of a problem for which the Schwarz method of P. L. Lions does not converge is the Helmholtz boundary value problem [40]. Bruno Després was the first to successfully apply domain decomposition method to the Helmholtz equation [33]. His approach consisted of applying Robin type transmission conditions with complex coefficients. The convergence of the resulting method however, is not satisfactory [40]. That has prompted many researchers to consider modifications of the method utilizing improved transmission operators. Nevertheless, the work of B. Després is seminal, and constitutes the foundation of the algorithms presented in this thesis.

Improvements of the domain decomposition methods are concerned with the optimal choice of transmission conditions. Given the transmission condition of the form $\partial_n u + \Lambda u$ what is the choice of Λ that guarantees convergence and results in a fast method? The transmission operator Λ could be a constant, local or non-local. For a large class of second-order problems and decompositions of the rectangular domains into strips, it has been shown that the optimal choice of Λ corresponds the DtN operator [62]. Similar results have been confirmed for the Helmholtz equation [69,70]. Unfortunately, DtN operators are in general non-local and therefore too costly to use in an efficient numerical algorithm. In the context of the Helmholtz problem, a great variety of techniques based on local transmission conditions have been proposed to improve the convergence. These include the optimized Schwarz method approach [40] and the evanescent mode damping algorithm [16, 21, 23]. The localized transmission operators however, do not accurately approximate the exact DtN operator on all of the modes of the solution. This results in a sub-optimal iterative method. In this thesis, we use and adapt current state of the art non-overlapping domain decomposition techniques based on localization of the DtN operator by a Padé series [22]. The rate of convergence of such methods have been shown to be optimal on the evanescent modes and significantly improved for the remaining modes. As we will show, our algorithms coupling finite elements with perfectly matched layer

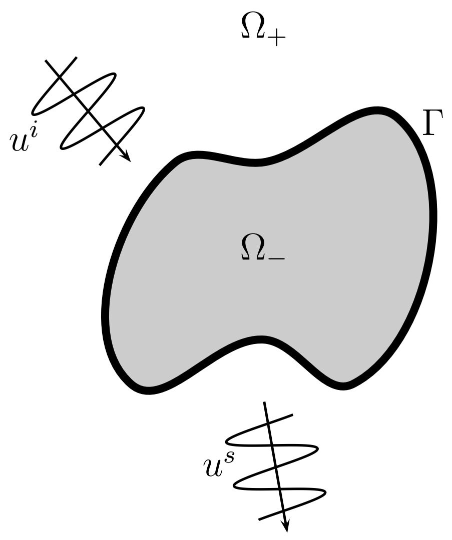


Figure 2.1 Problem domain.

technique results in an efficient domain decomposition method that exhibits good convergence properties independent of the wavenemuber.

2.1 Domain Decomposition Method for a 2D Model Problem

For the sake of clarity, let us consider a typical two dimensional problem of acoustic scattering from a sound hard obstacle. The problem domain (Figure 2.1) consists of an obstacle which is a bounded subset Ω_- of the Euclidean space $\mathbb{R}^2$ with boundary Γ. For simplicity we assume that the obstacle is impenetrable in which case the wave propagation takes place only in the exterior region $\Omega_+ = \mathbb{R}^2 \setminus \Omega_-$.

Given the amplitude of the incoming plane wave u^i, the problem for the scattered field u^s we want to solve is as follows (see Section 1.2):

$$\begin{cases} \Delta u^s + k^2 u^s = 0 \quad \text{in} \quad \Omega_+ \\ \partial_\nu u^s = -\partial_\nu u^i \quad \text{on} \quad \Gamma \\ \lim_{r\to\infty} r^{\frac{1}{2}} \left(\frac{\partial u^s}{\partial r} - iku^s\right) = 0 \quad \text{for} \quad r = |\mathbf{x}| \end{cases} \tag{2.1}$$

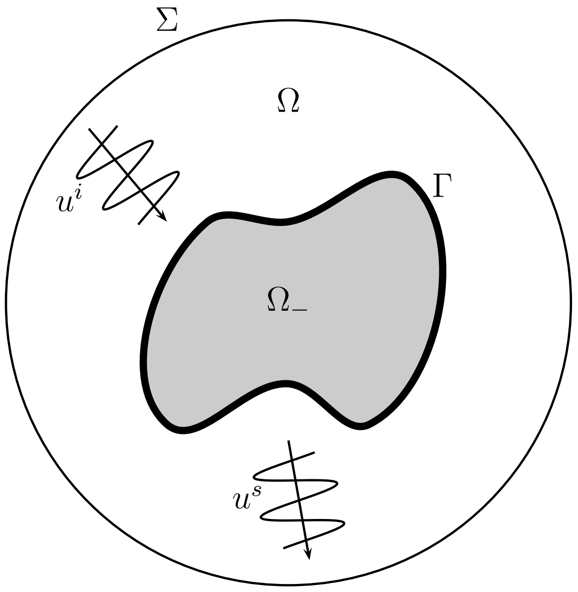


Figure 2.2 Truncated problem domain.

where ν is the unit normal vector to the boundary Γ that points into Ω_-. For the numerical purposes, we truncate the original problem domain (see Figure 2.2) with a circular boundary Σ on which we specify an ABC.

Provided that Σ is sufficiently far away from the scatterer, a good approximation to the Dirichlet-to-Neumann map on that curve is $\partial_r u = iku$ [61]. Using this relations as our ABC, we arrive at the following problem:

$$\begin{cases} \Delta u^s + k^2 u = 0 \quad \text{in} \quad \Omega \\ \partial_\nu u^s = -\partial_\nu u^i \quad \text{on} \quad \Gamma \\ \partial_r u^s - iku^s = 0 \quad \text{on} \quad \Sigma \end{cases} \tag{2.2}$$

It should be pointed out that we have slightly abused notation here. The solution u^s of the above problem is not the same as the solution to the original, infinite problem (2.1). Solution u^s to problem (2.2) contains truncation error due to an ABC not present in the original problem. Nevertheless, in what follows, this should not cause any confusion.

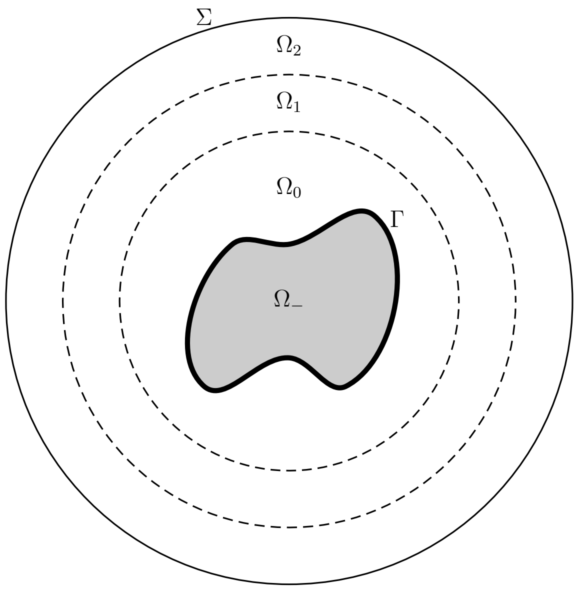


Figure 2.3 Decomposition without cross-points.

2.2 Formulation of the Domain Decomposition Method.

In order to solve problem (2.2) using domain decomposition approach, we partition the computational domain Ω into a disjoint collection of N_d subdomains $\Omega_i : i = 0, 1, \ldots, N_d - 1$ such that:

- $\overline{\Omega} = \bigcup_{i=0}^{Nd-1} \overline{\Omega}_i$
- $\Omega_i \cap \Omega_j = \emptyset$ if $i \neq j$
- $\partial\Omega_i \cap \partial\Omega_j = \overline{\Sigma}_{ij} = \overline{\Sigma}_{ji}$ is the artificial interface separating Ω_i from Ω_j.

A typical decomposition of the computational domain is depicted in the Figure (2.3) below. Let us denote by u_i a solution to the problem on the subdomain Ω_i and by ν_i the unit normal vector to the boundary $\partial\Omega_i$ of the subdomain Ω_i pointing away from it. Also, let $\Gamma_i = \partial\Omega_i \cap \Gamma$ and $\Sigma_i = \partial\Omega_i \cap \Sigma$ be those portions of $\partial\Omega_i$ that lie on the physical boundaries. Note that these sets may be empty for some subdomains. We will also introduce a set of subdomain indices $\sigma_i \subset \{0, 1, \ldots, N_d - 1\}$ such that if $j \in \sigma_i$ then Ω_j is a neighbor of Ω_i. Now consider the following system of coupled

Helmholtz boundary value problems:

$$\begin{cases} \Delta u_i + k^2 u_i = 0 \quad \text{in} \quad \Omega_i \\ \partial_{\nu_i} u_i = \partial_{\nu_i} u^i \quad \text{on} \quad \Gamma_i \\ \partial_{\nu_i} u_i - iku_i = 0 \quad \text{on} \quad \Sigma_i \end{cases} \tag{2.3}$$

supplemented with continuity conditions at the artificial interfaces $\Sigma_{ij} : j \in \sigma_i$:

$$u_i = u_j \quad \text{and} \quad \partial_{\nu_i} u_i = -\partial_{\nu_j} u_j. \tag{2.4}$$

Following the approach of P. L. Lions [58] and B. Després [33] we begin by combining the continuity conditions (2.4) in the following equivalent form:

$$\partial_{\nu_i} u_i + \Lambda u_i = -\partial_{\nu_j} u_j + \Lambda u_j, \tag{2.5}$$

$$\partial_{\nu_i} u_j + \Lambda u_j = -\partial_{\nu_i} u_i + \Lambda u_i \tag{2.6}$$

where Λ is the transmission operator which for generality, is currently left unspecified. The original choice of Λ made by Després in [33] was $\Lambda = -ik$. Finally, let $n \geq 0$ be the iteration index. The iteration procedure consists of solving following problems at each step:

$$\begin{aligned} \Delta u_i^{(n+1)} + k^2 u_i^{(n+1)} &= 0 \quad \text{in} \quad \Omega_i \\ \partial_{\nu_i} u_i^{(n+1)} &= \partial_{\nu_i} u^i \quad \text{on} \quad \Gamma_i \\ \partial_{\nu_i} u_i^{(n+1)} - iku_i^{(n+1)} &= 0 \quad \text{on} \quad \Sigma_i \end{aligned} \tag{2.7}$$

$$\partial_{\nu_i} u_i^{(n+1)} + \Lambda u_i^{(n+1)} = -\partial_{\nu_j} u_j^{(n)} + \Lambda u_j^{(n)} \quad \text{for} \quad \Sigma_{ij} : j \in \sigma_i. \tag{2.8}$$

Introduction of an iterative scheme allowed us to relax the continuity conditions (2.5)-(2.6) and close the local problems with the transmission condition (2.8). The fact that the interface condition depends on the solution to a neighboring problem at a previous iteration step enables the algorithm to solve the local problems concurrently. This single feature forms the basis for a natural parallel implementations of the method. Let us now define the interface datum:

$$g_{ij}^{(n)} = -\partial_{\nu_j} u_j^{(n)}|_{\Sigma_{ij}} + \Lambda u_j^{(n)}|_{\Sigma_{ij}}. \tag{2.9}$$

At the $(n+1)^{\text{th}}$ step, the problem (2.7)-(2.8) is solved with interface data $g_{ij}^{(n)} : j \in \sigma_i$. Once the solution $u_i^{(n+1)}$ is obtained, the interface data $g_{ji}^{(n+1)} = -\partial_{\nu_i} u_i^{(n)}|_{\Sigma_{ij}} + \Lambda u_i^{(n)}|_{\Sigma_{ij}}$ is calculated and communicated to the neighboring subdomains $\Omega_j : j \in \sigma_i$. This updates the iterative scheme and exchanges the data between neighboring subdomains. The calculation of the update may be problematic from the numerical point of view as it requires the determination of the normal derivative of the solution along the interface. A simple algebraic manipulation however removes the need for such a procedure:

$$\begin{aligned} g_{ij}^{(n+1)} &= \partial_{\nu_j} u_j^{(n+1)}|_{\Sigma_{ij}} + \Lambda u_j^{(n+1)}|_{\Sigma_{ij}} \\ &= -\partial_{\nu_j} u_j^{(n+1)}|_{\Sigma_{ij}} + \Lambda u_j^{(n+1)}|_{\Sigma_{ij}} + \Lambda\, u_j^{(n+1)}|_{\Sigma_{ij}} - \Lambda\, u_j^{(n+1)}|_{\Sigma_{ij}} \\ &= -g_{ji}^{(n)} + 2\Lambda\, u_i^{(n+1)}|_{\Sigma_{ij}}. \end{aligned} \tag{2.10}$$

2.3 Discussion of the Convergence of DDM

Improving the convergence properties of the iterative process constitutes the key in designing effective domain decomposition algorithms. It is well known that the convergence of the domain decomposition methods for scattering problems strongly depends on the choice of the transmission operator Λ. Indeed, to each choice of Λ corresponds an *iteration operator* with particular spectral properties [22]. The optimal convergence is obtained by defining the transmission conditions on each artificial interface using the Dirichlet-to-Neumann (DtN) map [40, 69, 70]. This however leads to a very expensive procedure in practice as the DtN operator is generally non local. Therefore, improved techniques usually introduce modifications to the original algorithm developed by B. Després that rely on more accurate local representation of the DtN operator. Indeed, and not incidentally, the original choice of $\Lambda = -ik$ made by B. Després, corresponds to a low-order approximation of that operator. However, in [16] it is shown that the resulting iteration operator acts on the part of the spectrum corresponding to the propagating modes, while the eigenvalues related to the evanescent modes have unit modulus. This directly impacts the convergence properties of the resulting iterative scheme.

Three families of techniques have been proposed to overcome this problem. First, algorithms based on the optimization of the rate of convergence were introduce by Gander et al. [39], where improved local approximations of the DtN map of order zero and order two are built. For a generic interface Σ, those approximations take, respectively, the following form:

$$\Lambda\, u = \alpha u, \quad \text{and} \quad \Lambda\, u = \alpha u + \Delta_\Sigma u \tag{2.11}$$

where Δ_Σ is the Laplace-Beltrami operator on Σ and the complex coefficients α and β are obtained by solving a min-max optimization problem on the rate of convergence. Second, the Evanescent Modes Damping Algorithm (EMDA) was introduce by Boubendir et. al [23], with the explicit aim to damp the evanescent modes. This method achieves the improvement on the evanescent modes by applying the following transmission operator:

$$\Lambda\, u = -iku + \chi u \tag{2.12}$$

where χ is a self-adjoint positive operator. Finally, the third technique, also proposed by Boubendir et. al [22] utilizes the so called *square-root* transmission operator:

$$\Lambda\, u = -ik\sqrt{1 + \nabla_\Sigma \cdot (k_\epsilon^{-2}\nabla_\Sigma)}\, u \tag{2.13}$$

where $k_\epsilon = k + i\epsilon$ is a complexified wavenumber. The operator $\nabla_\Sigma\cdot$ is the surface divergence of a tangent vector field on Σ and ∇_Σ is the tangential gradient of the surface field. The square root $\sqrt{A}$ of an operator A is classically defined through the spectral decomposition of A [73]. In the remainder of this section, we shall give an outline of the derivation of the non-local operator in Equation (2.13) and present a localization procedure based on Padé approximates.

To present the derivation of (2.13) we begin with a formal construction of the DtN operator for a half-plane. To this end consider the following boundary value problem:

$$\begin{cases} \Delta u + k^2 u = 0 \quad \text{in} \quad \mathbb{R}^3_+ = \{\mathbf{x} \in \mathbb{R}^3 : \, x_1 > 0\} \\ u = g \quad \text{on} \quad \Sigma \\ u \quad \text{is outgoing} \end{cases} \tag{2.14}$$

where the boundary $\Sigma := \{\mathbf{x} \in \mathbb{R}^3 : \, x_1 = 0\}$. Therefore, the standard basis vector $\mathbf{e}_1$ gives the normal direction to Σ while vectors $\mathbf{e}_2$ and $\mathbf{e}_3$ are tangential to it. We denote by $\mathcal{D}$ the DtN operator defined by [22]:

$$\begin{aligned} &\mathcal{D} : H^{1/2}(\Sigma) \to H^{-1/2}(\Sigma), \\ &u|_\Sigma \mapsto \partial_{x_1} u|_\Sigma = \mathcal{D}(u|_\Sigma). \end{aligned} \tag{2.15}$$

The simplicity of the problem geometry in (2.14) makes it possible to write $\mathcal{D}$ explicitly by Fourier analysis. Indeed, let us introduce $\boldsymbol{\xi} = (\xi_2, \xi_3) \in \mathbb{R}^2$ as the Fourier covariable of the tangential variable $\mathbf{x}_\perp = (x_2, x_3)$. We denote by $\mathcal{F}_{\mathbf{x}_\perp}$ the partial Fourier transform with respect to (x_2, x_3) and by $\mathcal{F}^{-1}_{\boldsymbol{\xi}}$ the associated inverse Fourier transform. Applying $\mathcal{F}_{\mathbf{x}_\perp}$ to the Helmholtz equation (2.14) leads to an ODE:

$$\partial^2_{x_1} \hat{u}(x_1, \boldsymbol{\xi}) + (k^2 - |\boldsymbol{\xi}|^2)\, \hat{u}(x_1, \boldsymbol{\xi}) = 0 \tag{2.16}$$

for $x_1 \geq 0$ where $\hat{u}$ denotes the transform of u. Let us define:

$$\sigma_1(k, \boldsymbol{\xi}) = ik\sqrt{1 - |\boldsymbol{\xi}|^2/k^2}. \tag{2.17}$$

The solution to the above equations can now be written as:

$$\hat{u}(x_1, \boldsymbol{\xi}) = A^{+} e^{\sigma_1(k,\boldsymbol{\xi})\, x_1} + A^{-} e^{-\sigma_1(k,\boldsymbol{\xi})\, x_1}. \tag{2.18}$$

Now, since we are looking for an outgoing solution to (2.14) we need to choose coefficients A^+ and A^- so that the $L^2(\mathbb{R}^3_+)$-norm of u is finite. By Parseval's relation, this also means that the $L^2(\mathbb{R}^3_+)$-norm of $\hat{u}$ is finite. This can only arise if $A^- = 0$ resulting in the right traveling solution $\hat{u}(x_1, \boldsymbol{\xi}) = A^+ e^{\sigma_1(k,\boldsymbol{\xi})\, x_1}$. Deriving the last expression with respect to x_1, applying the inverse Fourier transform in $\boldsymbol{\xi}$ and considering the trace of the resulting relation on the transmitting boundary Σ leads to:

$$\partial_{x_1} u(0, \mathbf{x}_\perp) = \mathcal{F}^{-1}_{\boldsymbol{\xi}} \left\{ \sigma_1(k, \boldsymbol{\xi}) \hat{u}(x_1, \boldsymbol{\xi}) \right\} |_{\Sigma}. \tag{2.19}$$

In the pseudo-differential operator terminology the quantity $\sigma_1(k, \boldsymbol{\xi})$ is referred to as the symbol of the corresponding operator $\mathcal{D}$. This relationship between $\mathcal{D}$ and its symbol is written as $\mathcal{D} = \mathrm{Op}(\sigma_1)$. Since $\sigma_1(k, \boldsymbol{\xi})$ is not a polynomial in $\boldsymbol{\xi}$, $\mathcal{D}$ is not a differential operator. Instead, $\mathcal{D}$ belongs to a more general class of operators called pseudo-differential operators [73]. It is shown in [22] that:

$$\mathcal{D} = ik\sqrt{1 + \frac{\Delta_\Sigma}{k^2}} \tag{2.20}$$

where the Laplace-Beltrami operator over Σ is defined by: $\Delta_\Sigma := \partial^2_{x_2} + \partial^2_{x_3}$. For a half-space, the transmission operator Λ is thus simply taken to be equal to:

$$\Lambda u = -\mathcal{D} u = -ik\sqrt{1 + \frac{\Delta_\Sigma}{k^2}}\, u. \tag{2.21}$$

Consider now a curved surface Σ. The surface is locally approximated by its tangent plane, and one can formally propose an approximate representation of Λ over Σ based on (2.21) where $\Delta_\Sigma = \nabla_\Sigma \cdot \nabla_\Sigma$ is the Laplace-Beltrami operator for the curved surface Σ.

The fact that Λ is no longer an exact DtN operator on Σ has been exemplified in the case of a circular geometry in [7] and more general considerations are presented in [5]. In particular, numerical experiments have shown that Λ is not uniform over all $\boldsymbol{\xi}$ and is not valid for the glancing rays corresponding to $|\boldsymbol{\xi}| \approx k$. This behavior can be attributed to the loss of analyticity by σ_1 in exactly that region. As pointed out in the reference, operator Λ can be regularized by adding a small imaginary number $i\epsilon$ to the wavenumber k. As will be seen shortly, ϵ depends on both the wavenumber k as well as the local curvature of the surface. The symbol of Λ is redefined to:

$$\sigma_{1,\epsilon}(k, \boldsymbol{\xi}) = -ik\sqrt{1 - |\boldsymbol{\xi}|^2/k_\epsilon^2} \tag{2.22}$$

where $k_\epsilon = k + i\epsilon$ for $\epsilon > 0$. The newly obtained operator [7, 22]:

$$\mathrm{Op}(\sigma_{1,\epsilon}) = -ik\sqrt{1 + \nabla_\Sigma \cdot \left(\frac{1}{k_\epsilon^2}\nabla_\Sigma\right)} \tag{2.23}$$

is the operator Λ of Equation (2.13). In the case of circular surface Σ of radius R the optimal value of $\epsilon = \epsilon_{\text{opt}}$ is chosen to minimize the magnitude of the reflection coefficients resulting in $\epsilon_{\text{opt}} \approx 0.4k^{1/3}R^{-2/3}$ [7]. For a general boundary Σ with local curvature κ the optimal value of ϵ is formally set to:

$$\epsilon_{\text{opt}} \approx 0.4k^{1/3}\kappa^{2/3}. \tag{2.24}$$

2.3.1 Localization of the Square Root Operator by Complex Padé Series

The square root operator given by Equation (2.23) is a non-local operator. Therefore, it is impractical in a finite element setting since the discretization would lead to full matrices for the interface unknowns. Fortunately, this operator can be efficiently approximated via a series of local, partial differential operators resulting in linear systems with sparse matrices. This localization process has been presented in [7, 22] and is realized by a rotating branch-cut approximation of the square root function, and a subsequent application of complex Padé approximation of order N_p:

$$\begin{aligned}\sqrt{1+\nabla_\Sigma\cdot\left(\frac{1}{k_\epsilon^2}\nabla_\Sigma\right)} &\approx R^\alpha_{N_p}\left(\nabla_\Sigma\cdot\left(\frac{1}{k_\epsilon^{-2}}\nabla_\Sigma\right)\right)\,u \\ &= C_0u+\sum_{l=1}^{N_p}A_l\nabla_\Sigma\cdot\left(k_\epsilon^{-2}\nabla_\Sigma\right)\left(1+B_l\nabla_\Sigma\cdot\left(k_\epsilon^{-2}\nabla_\Sigma\right)\right)^{-1}u\end{aligned} \tag{2.25}$$

which corresponds to the complex Padé approximation:

$$\sqrt{1+z}\approx R^\alpha_{N_p}(z)=C_0+\sum_{l=1}^{N_p}\frac{A_lz}{1+B_lz} \tag{2.26}$$

The complex Padé coefficients A_l, B_l, C_0 are given by:

$$C_0=e^{i\frac{\alpha}{2}}R_{N_p}\left(e^{-i\alpha}-1\right),\quad A_l=\frac{e^{-i\frac{\alpha}{2}}\,a_l}{(1+b_l(e^{-i\alpha}-1))^2},\quad B_l=\frac{e^{-i\frac{\alpha}{2}}\,b_l}{1+b_l(e^{-i\alpha}-1)}$$

where α is the angle of rotation, a_l, b_l are the standard Padé coefficients:

$$a_l=\frac{2}{2N_p+1}\sin^2\left(\frac{l\pi}{2N_p+1}\right),\qquad b_l=\cos^2\left(\frac{l\pi}{2N_p+1}\right)$$

and finally $R_{N_p}=1+\sum_{l=1}^{N_p}\frac{a_lz}{1+b_lz}$ is the real Padé approximation of order N_p to $\sqrt{1+z}$.

Auxiliary Functions. Given the relation (2.21) and Equation (2.25) the localized transmission operator Λ over a transmitting surface Σ takes the form:

$$\Lambda u\approx -ik\left(C_0u+\sum_{l=1}^{N_p}A_l\nabla_\Sigma\cdot\left(k_\epsilon^{-2}\nabla_\Sigma\right)\left(1+B_l\nabla_\Sigma\cdot\left(k_\epsilon^{-2}\nabla_\Sigma\right)\right)^{-1}\right)u. \tag{2.27}$$

The variational formulation of local problems (2.7) - (2.8) with the transmission operator given above is more convenient to obtain with an introduction of auxiliary functions φ_l. Those surface functions are used to rewrite Equation (2.27) so that it does not contain inverse operators:

$$\Lambda u \approx -ik\left(C_0 u + \sum_{l=1}^{N_p} A_l \nabla_\Sigma \cdot \left(k_\epsilon^{-2} \nabla_\Sigma \varphi_l\right)\right). \tag{2.28}$$

where the functions $\varphi_l : l = 1, \ldots, N_p$ are defined on Σ as the solution the following surface PDEs:

$$\left(1 + B_l \nabla_\Sigma \cdot \left(k_\epsilon^{-2} \nabla_\Sigma\right)\right) \varphi_l = u. \tag{2.29}$$

The local, continuous problems (2.7) - (2.8) need to be modified accordingly to accommodate the new transmission operator Λ given in Equation (2.28):

$$\begin{aligned}
\Delta u_i^{(n+1)} + k^2 u_i^{(n+1)} &= 0 \quad \text{in} \quad \Omega_i \\
\partial_{\nu_i} u_i^{(n+1)} &= \partial_{\nu_i} u^i \quad \text{on} \quad \Gamma_i \qquad (2.30)\\
\partial_{\nu_i} u_i^{(n+1)} - iku_i^{(n+1)} &= 0 \quad \text{on} \quad \Sigma_i
\end{aligned}$$

$$\partial_{\nu_i} u_i^{(n+1)} - ik\left(C_0 u_i^{(n+1)} + \sum_{l=1}^{N_p} A_l \nabla_{\Sigma_{ij}} \cdot \left(k_\epsilon^{-2} \nabla_{\Sigma_{ij}} \varphi_{i(l)}^{(n+1)}\right)\right) = g_{ij}^{(n)} \tag{2.31}$$

$$\left(1 + B_l \nabla_{\Sigma_{ij}} \cdot \left(k_\epsilon^{-2} \nabla_{\Sigma_{ij}}\right)\right) \varphi_{i(l)}^{(n+1)} = u_{i(l)}^{(n+1)} \quad \Sigma_{ij} : j \in \sigma_i, \quad l = 1, \ldots, N_p \tag{2.32}$$

where:

$$g_{ij}^{(n)} = -\partial_{\nu_j} u_j^{(n)} - ik\left(C_0 u_j^{(n)} + \sum_{l=1}^{N_p} A_l \nabla_\Sigma \cdot \left(k_\epsilon^{-2} \nabla_{\Sigma_{ij}} \varphi_{j(l)}^{(n)}\right)\right) \quad \text{for} \quad \Sigma_{ij} : j \in \sigma_i \tag{2.33}$$

The scheme is updated according to:

$$g_{ij}^{(n+1)} = -g_{ji}^{(n)} - 2ik\left(C_0 u_j^{(n+1)} + \sum_{l=1}^{N_p} A_l \nabla_\Sigma \cdot \left(k_\epsilon^{-2} \nabla_{\Sigma_{ij}} \varphi_{j(l)}^{(n+1)}\right)\right) \quad \text{for} \quad \Sigma_{ij} : j \in \sigma_i \tag{2.34}$$

Finite element approximation of the resulting coupled problem for the local solution u as well as the auxiliary functions φ is are presented the next section. More details can be found in Appendix A.

2.4 Finite Element Discretization

Solution to the problem (2.7)-(2.8) as well as (2.30) - (2.32) can be conveniently approximated in the framework of the nodal finite element method [25, 48]. In particular, we use triangular finite elements. To this end we introduce triangulation T^h of the computational domain Ω where parameter h denotes the mesh size. We assume that T^h induces conforming triangulations of $\Omega_i : i = 0, 1, \dots, N_d - 1$ which we denote by Ω_i^h. We let X_i^h to be the finite dimensional subspace of the solution space $X_i = H^1(\Omega_i)$ consisting of piecewise linear polynomials. By u_i^h we denote the approximation to $u_i^{(n+1)}$ in X_i^h and g_{ij}^h is the projection of $g_{ij}^{(n)}$ onto that space. The finite element method works with variational formulation of the problem which in our case can be stated as follows. Find $u_i^h \in X_i^h$ such that:

$$a_i^h\left(u_i^h, v_i^h\right) + \sum_{j\in\sigma_i} c_{ij}^h\left(\Lambda u_i^h, v_i^h\right) = l_i^h\left(v_i^h\right) + \sum_{j\in\sigma_i} c_{ij}^h\left(g_{ij}^h, v_i^h\right) \tag{2.35}$$

for all $v_i^h \in X_i^h$. The functionals a_i^h, l_i^h and c_{ij}^h are defined by:

$$\begin{aligned} a_i^h\left(u_i^h, v_i^h\right) &= \int_{\Omega_i^h} \nabla u_i^h \cdot \nabla v_i^h \, dA - k^2 \int_{\Omega_i^h} u_i^h v_i^h \, dA - ik \int_{\Sigma_i^h} u_i^h v_i^h \, ds \\ l_i^h\left(v_i^h\right) &= \int_{\Gamma_i^h} g v_i^h \, ds \\ c_{ij}^h\left(u_i^h, v_i^h\right) &= \int_{\Sigma_{ij}^h} u_i^h v_i^h \, ds \end{aligned} \tag{2.36}$$

In the above Σ_i^h, Γ_i^h and Σ_{ij}^h denote respectively the piecewise linear approximation to curves Σ_i, Γ_i and Σ_{ij} induced by the triangulation T^h. In particular, for Λ defined by Equation (2.28), the discrete variational problem (2.35) obtained with help of integration by parts, takes the form:

$$a_i^h\left(u_i^h, v_i^h\right) - ikC_0 \sum_{j\in\sigma_i} c_{ij}^h\left(u_i^h, v_i^h\right) + ik \sum_{j\in\sigma_i}\sum_{l=1}^{N_p} A_l\, c_{ij}^h\left(k_\epsilon^{-2}\nabla_{\Sigma_{ij}}\varphi_{i(l)}^h, \nabla_{\Sigma_{ij}} v_i^h\right) = l_i^h\left(v_i^h\right) + \sum_{j\in\sigma_i} c_{ij}^h\left(g_{ij}^h, v_i^h\right) \tag{2.37}$$

$$c_{ij}^h\left(\varphi_{i(l)}^h, v_i^h\right) - B_l\, c_{ij}^h\left(k_\epsilon^{-2}\nabla_{\Sigma_{ij}}\varphi_{i(l)}^h, \nabla_{\Sigma_{ij}} v_i^h\right) = c_{ij}^h\left(u_i^h, v_i^h\right) \quad j\in\sigma_i, \quad l=1,\dots,N_p \tag{2.38}$$

Evaluating (2.35) at the Lagrange basis for X_i^h yields a system of equations that needs to be solved at each iteration. The system matrix however can be factorized during the pre-processing stage using an efficient sparse LU factorization method [56]. At each step, the solution can then be obtained by a simple and inexpensive back-substitution procedure. More details concerning the application of finite element method to the Helmholtz problem can be found in Appendix A.

CHAPTER 3

PRECONDITIONING THE PERFECTLY MATCHED LAYER SYSTEM

3.1 Motivation

To overcome the difficulties associated with the standard ABCs, an alternative approach to deal with the truncation of unbounded domains was introduced by Bérenger [17] in the context of Maxwell's equations in electromagnetism. His method consisted of surrounding the computational domain with a thin layer of an artificial material designed to absorb the scattered field radiated from the obstacle. This method is referred to as *perfectly matched* because the interface between the domain of interest and the absorbing layer is transparent to the propagating waves. Though initially settled for Maxwell's equations, the PML idea has also been applied to wave problems in acoustics [46, 47], elasticity [31] and shallow water waves [63]. In practice, the thickness of the layer is finite and kept small for computational efficiency. The boundary conditions specified on the truncation boundary of the PML seem to have little effect on the method's performance [30, 64]. As compared with local ABCs, discretization of the PML system leads to larger Hermitian matrices which are however still very sparse. This makes them a good candidate for iterative solvers. Unfortunately, the convergence rate of classical iterative methods applied to Helmholtz problems is slow [37]. One way to improve performance of such methods is to use a preconditioner. In this chapter we present a new preconditioner for Krylov methods obtained from zero frequency limit of the *complex frequency shifted* (CFS) version of the PML investigated in [14] by P.G. Petropoulos. The new preconditioner is easy to construct and shows great improvement in the convergence rate of GMRES iterations as compared with the diagonal and approximate inverse preconditioners [29].

Domain decomposition methods can be though of as preconditioned iterative methods [74]. It is therefore natural to investigate the PML problem in the context of DDM. We shall shall investigate this approach in the following chapter.

3.2 PML Problem Formulation

Consider again the problem (1.19) whose geometry is depicted in in Figure (2.1). The application of the PML technique to the numerical solution of that problem involves enclosing the scatterer by a thin layer as shown below in Figure (3.1).

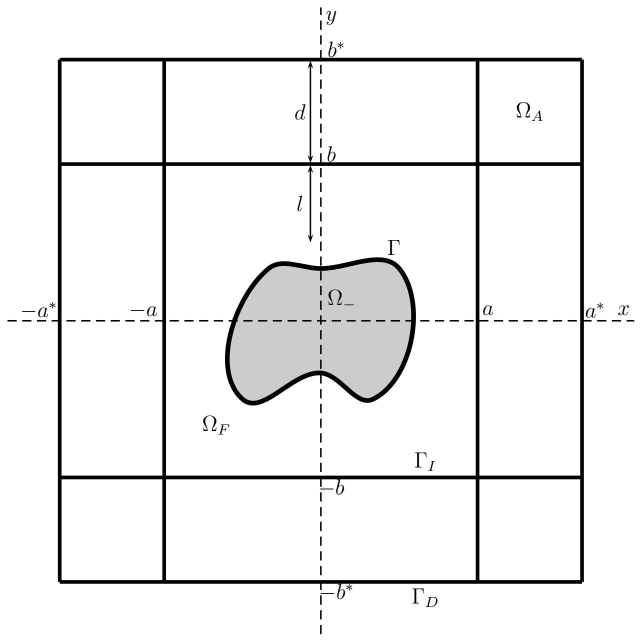


Figure 3.1 Problem domain truncated with PML.

The computational region obtained in the process is denoted by Ω_F. The PML region is denoted by Ω_A and it occupies the region of space between the PML interface Γ_I and domain boundary Γ_D. Its thickness is denoted by d and the minimum distance from the obstacle by l. Derivation of PML equations is now well known [48, section 3.3.4]. Following the notation used in the reference, we denote by u_F^s and

u_A^s the amplitudes of the scattered pressure waves in the physical and in the PML regions, respectively. These two coupled quantities satisfy:

$$\begin{cases} \Delta u_F^s + k^2 u_F^s = 0 \quad \text{in} \quad \Omega_F \\ \gamma_x^{-1}\partial_x\left(\gamma_x^{-1}\partial_x u_A^s\right) + \gamma_y^{-1}\partial_y\left(\gamma_y^{-1}\partial_y u_A^s\right) + k^2 u_A^s = 0 \quad \text{in} \quad \Omega_A \\ \partial_\nu u_F^s = -\partial_\nu u^i \quad \text{on} \quad \Gamma \\ u_F^s = u_A^s \quad \text{on} \quad \Gamma_I \\ \partial_\nu u_F^s = -\partial_\nu u_A^s \quad \text{on} \quad \Gamma_I \\ u_A^s = 0 \quad \text{on} \quad \Gamma_D \end{cases} \tag{3.1}$$

where the coefficient functions are:

$$\begin{aligned} \gamma_x(x) &= \begin{cases} 1 & x < a, \\ 1 + \frac{i}{k}\sigma_x(|x|) & a \le |x| < a^* \end{cases} \\ \gamma_y(x) &= \begin{cases} 1 & y < b, \\ 1 + \frac{i}{k}\sigma_y(|y|) & b \le |y| < b^* \end{cases} \end{aligned} \tag{3.2}$$

with ν is the vector normal to Γ_I pointing away from Ω_F.

The performance of PML is greatly affected by the choice of the absorbing functions σ_x and σ_y. Firstly, to avoid discontinuity in the equation coefficients, both are set to zero at the PML interface. Furthermore, simple Fourier mode analysis of the related half plane problem shows that the reflection from the terminating boundary is minimized provided that both σ_x and σ_y are monotonically increasing and achieve possibly large maximum values at that boundary [18,30]. Popular choice is the quadratic form of the absorbing functions:

$$\sigma_x(x) = \sigma^*(x-a)^2 \qquad \text{and} \qquad \sigma_y(y) = \sigma^*(y-b)^2 \tag{3.3}$$

where σ^* is a constant. Moreover, the PML parameters l, d, σ^* and γ are not arbitrary. Numerical studies indicate the existence of optimal parameter values which result in minimal error. Such optimal values depend on the problem data as well as on the mesh size and consequently are difficult to obtain. No theoretical procedure is yet know to estimate them, although some efforts towards it have been made in [30].

3.3 Frequency Shifted PML

Substituting γ_x and γ_y into the PML Equation (B.21) we obtain:

$$\begin{aligned}\left(1+\frac{i}{k}\sigma_x\right)^{-1}\partial_x\left(\left(1+\frac{i}{k}\sigma_x\right)^{-1}\partial_x u_A\right)+&\\ +\left(1+\frac{i}{k}\sigma_y\right)^{-1}\partial_y\left(\left(1+\frac{i}{k}\sigma_y\right)^{-1}\partial_y u_A\right)+k^2u_A=0&\end{aligned} \tag{3.4}$$

Now it is easy to see that the above equations does not correctly reduce to a Laplace equation in stretched coordinates as $k \to 0$. Some of the problems associated with this formulation are discussed in [65] by P.G. Petropoulos. To recover the correct limiting form of the equation, the author of the reference introduces a new small parameter γ and redefines the absorbing functions as follows:

$$\gamma_x(x)=\begin{cases}1 & x<a,\\ 1+\frac{i}{k+i\gamma}\sigma_x(|x|) & a\le|x|<a^*\end{cases}$$

$$\gamma_y(x)=\begin{cases}1 & y<b,\\ 1+\frac{i}{k+i\gamma}\sigma_y(|y|) & b\le|y|<b^*\end{cases}$$

The pressure field in the artificial layer now satisfies:

$$\left(1+\frac{\sigma_x}{\gamma}\right)^{-1}\partial_x\left(\left(1+\frac{\sigma_x}{\gamma}\right)^{-1}\partial_x u_A\right)+ \tag{3.5}$$

$$+\left(1+\frac{\sigma_y}{\gamma}\right)^{-1}\partial_y\left(\left(1+\frac{\sigma_y}{\gamma}\right)^{-1}\partial_y u_A\right)=0 \tag{3.6}$$

in the zero frequency limit. This equation will be used in next section to derive the new preconditioner presented in this chapter.

3.4 Variational Formulation and Finite Element Approximation

Variational formulation as well as finite element discretization of the PML system of equations are presented in Appendix (B.2). The corresponding variational problem can be written as:

$$a(u_F, v) + b(u_A, v) = l(v) \tag{3.7}$$

where:

$$\begin{aligned} a(u,v) &= \int_{\Omega_F} \nabla u \cdot \nabla v \, dx\, dy - k^2 \int_{\Omega_F} uv \, dx\, dy \\ b(u,v) &= \int_{\Omega_A} \frac{\gamma_y}{\gamma_x} \frac{\partial u}{\partial x}\frac{\partial v}{\partial x} \, dx\, dy + \int_{\Omega_A} \frac{\gamma_x}{\gamma_y} \frac{\partial u}{\partial y}\frac{\partial v}{\partial y} \, dx\, dy - k^2 \int_{\Omega_A} \gamma_x \gamma_y uv \, dx\, dy \\ l(v) &= \int_{\Gamma} gv \, ds \end{aligned} \tag{3.8}$$

The task is to find both, u_F and u_A in the appropriate function spaces (Appendix B.2) such that Equation (3.7) is satisfied for all test functions v.

A finite element approximation to u_F and u_A, denoted respectively by u_F^h and u_A^h, is obtained by an application of the Galerkin method to the problem (3.7). Discrete solutions u_F^h and u_A^h correspond to orthogonal projections of u_F and u_A onto a finite dimensional subspace X^h in the so called energy inner product (see Appendix B.3). The approximate solution to (B.31) consists of finding u_F^h and u_A^h such that:

$$a^h(u_F^h, v^h) + b^h(u_A^h, v^h) = l^h(v^h) \tag{3.9}$$

for all v^h and where a^h, b^h and l^h are the discrete versions of the functionals defined by Equations (3.8). Assuming that the nodal finite element basis for X^h is denoted by $\{\phi_i\}$, the finite element system can be written as:

$$\mathbf{K}\,\boldsymbol{\alpha} = \mathbf{f} \tag{3.10}$$

where $\mathbf{K}_{ij} = a^h(\phi_i, \phi_j) + b^h(\phi_i, \phi_j)$, $\boldsymbol{\alpha}_j = \alpha_j$ and $\mathbf{f}_j = l^h(\phi_j)$.

3.5 Preconditioner Derivation

As explained in [37], the solution to equation (3.10) is difficult to obtain by classical iterative solvers. One way to improve the convergence of these methods is to use a preconditioner $\mathbf{P}$ and solve instead the following, equivalent problem:

$$\mathbf{P}^{-1}\mathbf{K}\,\boldsymbol{\alpha} = \mathbf{P}^{-1}\mathbf{f} \tag{3.11}$$

The general idea behind matrix preconditioning is that if we solve the above system iteratively, the convergence will depend on the properties of the new matrix $\mathbf{P}^{-1}\mathbf{K}$ instead of those of $\mathbf{K}$. If matrix $\mathbf{P}$ is well chosen, equation (3.11) may be solved more rapidly than equation (3.10).

The matrix preconditioner that we propose here is obtained from $\mathbf{K}$, in the limit as $k \to 0$, that is:

$$\mathbf{P}_{ij} = \lim_{k\to 0}\mathbf{K}_{ij} = \lim_{k\to 0}\left(a^h(\phi_i,\phi_j) + b^h(\phi_i,\phi_j)\right) = a_0^h(\phi_i,\phi_j) + b_0^h(\phi_i,\phi_j) \tag{3.12}$$

where $a_0^h(\phi_i,\phi_j)$ and $b_0^h(\phi_i,\phi_j)$ are obtained from (3.13) using (3.5) respectively:

$$\begin{aligned} a_0^h(u,v) &= \int_{\Omega_F^h} \nabla u \cdot \nabla v \, dx\, dy \\ b_0^h(u,v) &= \int_{\Omega_A^h} \frac{1+\frac{1}{\gamma}\sigma_y}{1+\frac{1}{\gamma}\sigma_x}\frac{\partial u}{\partial x}\frac{\partial v}{\partial x}\, dx\, dy + \int_{\Omega_A^h} \frac{1+\frac{1}{\gamma}\sigma_x}{1+\frac{1}{\gamma}\sigma_y}\frac{\partial u}{\partial y}\frac{\partial v}{\partial y}\, dx\, dy \end{aligned} \tag{3.13}$$

Matrix $\mathbf{P}$ is real and symmetric, it possesses the same sparsity structure as $\mathbf{K}$ and is straightforward to obtain once the assembly routine for $\mathbf{K}$ is available. From here on now, we will refer to $\mathbf{P}$ as the zero frequency limit preconditioner and denote it by $\mathbf{P}_{\text{zfl}}$.

3.6 Discussion and Numerical Results

To study the performance of the proposed precenditioner, we consider a problem of scattering of plane acoustic waves from a sound hard disk centered at the origin with radius R which we denote by Ω. Let u be the unknown amplitude of the scattered pressure wave. The time harmonic problem for u is:

$$\begin{cases} \Delta u + k^2 u = 0 \quad \text{in} \quad \mathbb{R}^2 - \Omega \\ \partial_r u = ike^{-i\mathbf{k}\cdot\mathbf{r}} \quad \text{on} \quad \Gamma \\ \lim_{r\to+\infty} \sqrt{r}\,(\partial_r u - iku) = 0 \end{cases} \tag{3.14}$$

where $\mathbf{r} = (x, y)$, $r = |\mathbf{r}|$ and $\mathbf{k} = (k\cos\theta_i, k\sin\theta_i))$ with θ_i being the angle of incidence. For simplicity we set $\theta_i = 0$. Then the solution to this problem is given by the Fourier-Hankel expansion:

$$u(r,\theta) = \sum_{m=-\infty}^{\infty} i^{-m} \frac{J'_m(kR)}{H_m^{(1)\prime}(kR)} H_m^{(1)}(kr)e^{im\theta} \tag{3.15}$$

where J_m and $H_m^{(1)}$ are the Bessel and Hankel functions of the first kind respectively and the prime symbol denotes differentiation with respect to kr.

The computational problem domain truncated with PML is depicted in Figure (3.2) below:

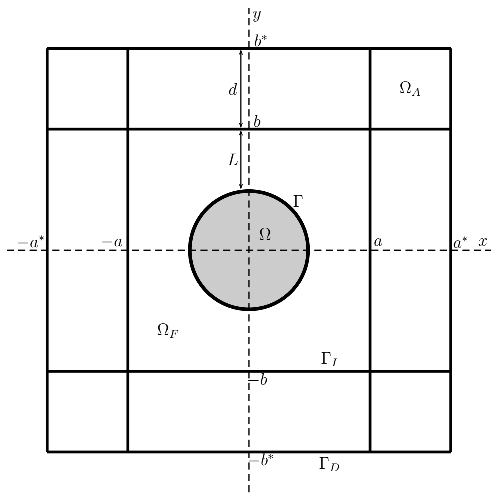


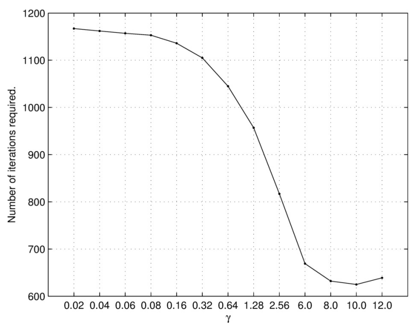
Figure 3.2 Problem domain truncated with PML.

In the remainder of this section we report the results of the numerical convergence study of GMRES iterations conducted by applying the new preconditioner $\mathbf{P}_{\text{zfl}}$ given by Equation (3.5), to the system (3.10). Problem parameters k, l, d, σ^* and γ are varied to show their effect on the error and the convergence profile. The number of points per wavelength is kept constant at 14. The performance of the preconditioner is compared with the inexpensive diagonal preconditioner as well as the approximate inverse preconditioner (AIP) described in [29] and also in [20] in the context of PML. We denote them by $\mathbf{P}_{\text{diag}}$ and $\mathbf{P}_{\text{aip}}$ respectively. The preconditioner $\mathbf{P}_{\text{diag}}$ is obtained by extracting the diagonal entries of the system matrix $\mathbf{K}$ whereas $\mathbf{P}_{\text{aip}}$ is found by minimizing the Frobenious norm of the matrix $\mathbf{I} - \mathbf{P}_{\text{aip}}^{-1}K$. The steepest descent algorithm used to approximate $\mathbf{P}_{\text{aip}}$ is quite expensive. Moreover, a numerical dropping strategy needs to be employed in order to ensure sparsity of $\mathbf{P}_{\text{aip}}$ which further compounds the computational cost of the algorithm. The numerical dropping in the search direction that we employ, also preserves the steepest descent property of the algorithm and is described in detail in [29].

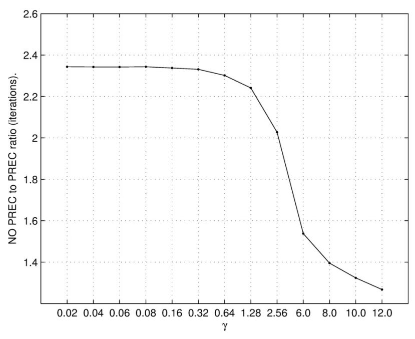
In an attempt to study the effect of the PML parameters on the performance of the proposed preconditioner we vary the values of γ and σ independently while keeping other problem parameters fixed. Measured in the units of wavelength, we set $R = 1.5$, $l = 1$, and $d = 1$ so that with a fixed wavenumber $k = \pi$ and 14 points per wavelength, the problem size stays the same through our numerical study.

The column headers of the tables below, indicate how the solution was obtained. DIRECT refers simply to finding $\boldsymbol{\alpha} = \mathbf{K}^{-1}\mathbf{f}$ by LU factorization, whereas NO PREC to applying GMRES to the finite element system $\mathbf{K}\boldsymbol{\alpha} = \mathbf{f}$. Similarly, DIAG PREC, ZFL PREC and AIP refer to an application of GMRES to $\mathbf{P}_{\text{diag}}^{-1}\mathbf{K}\boldsymbol{\alpha} = \mathbf{P}_{\text{diag}}^{-1}\mathbf{f}$, $\mathbf{P}_{\text{zfl}}^{-1}\mathbf{K}\boldsymbol{\alpha} = \mathbf{P}_{\text{zfl}}^{-1}\mathbf{f}$ and $\mathbf{P}_{\text{aip}}^{-1}\mathbf{K}\boldsymbol{\alpha} = \mathbf{P}_{\text{aip}}^{-1}\mathbf{f}$ respectively.

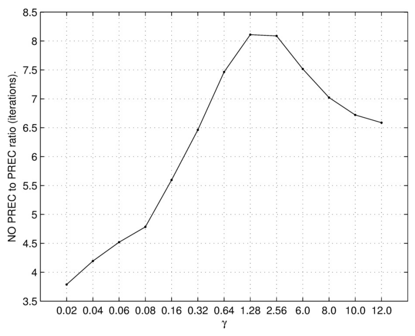
The Figure (3.3) shows the number of iterations needed for GMRES to attain the residual tolerance of 10^{-6}. Figure (3.3 a) shows this number as a function of the PML parameter γ for GMRES applied to an unpreconditioned system. The remaining three plots show the corresponding iteration counts as a fraction of those



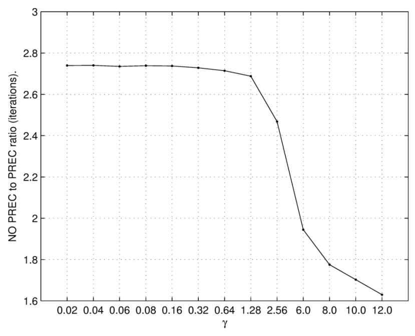
(a) No Preconditioner



(b) Diagonal Preconditioner



(c) ZFL Preconditioner



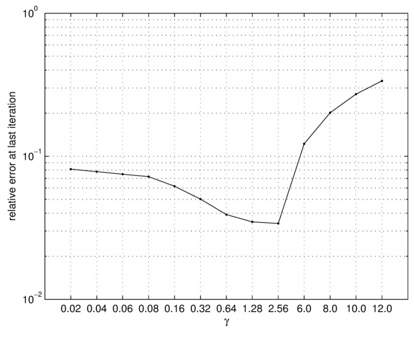
(d) Approximate Inverse Preconditioner

Figure 3.3 $\underline{\sigma = 50.0}$: number of iterations to attain residual tolerance of $1{\times}10^{-6}$.

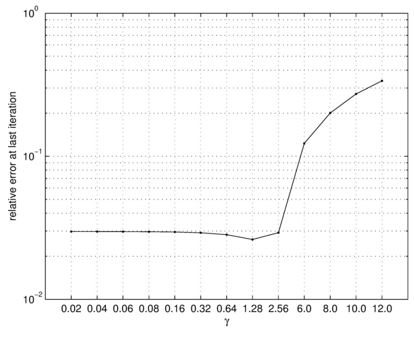
required by the unpreconditioned solver. Clearly, we see that the application of the ZFL preconditioner results in greatest reduction of required iterations. The most dramatic reduction in the number of iterations occurs at $\gamma = 1.28$ where the ZFL preconditioner lowers the iteration count by a factor of eight.

This particular value of γ appears to yet another significance. Looking at the accuracy of the four different solvers, we see that least relative error is achieved for $\gamma \approx 1.28$ as the Figure (3.4) shows. The value of $\sigma = 50.00$ have been chosen as it gave the least overall error.

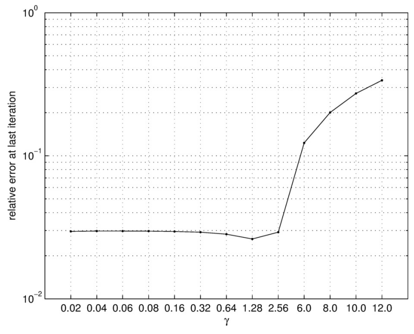
To understand the improved performance of the ZFL preconditioner we look at the modulus of the eigenvalues of the four PML systems for $\gamma \approx 1.28$ (Figures 3.5 - 3.7).



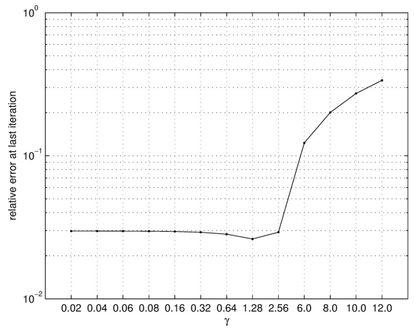
(a) No Preconditioner



(b) Diagonal Preconditioner



(c) ZFL Preconditioner



(d) Approximate Inverse Preconditioner

Figure 3.4 $\underline{\sigma = 50.0}$: relative error at last iteration

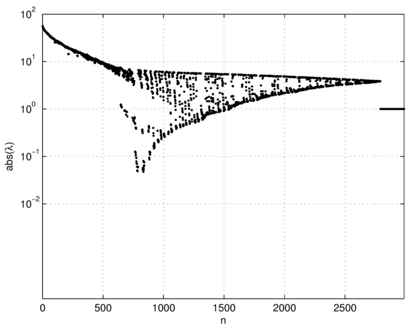
From the point of view of the GMRES, the zero frequency limit preconditioner performs best because it seems to improve the distribution of eigenvalues of the system matrix. We recall [72], that in the case of diagonalizable matrix $\mathbf{A} = \mathbf{X}\mathbf{D}\mathbf{X}^{-1}$ the residual norm at the m^{th} step of GMRES satisfies:

$$||r_{m+1}|| \leq \kappa(\mathbf{X}) \min_{p \in P_m, p(0)=1} \max_{\lambda_i \in \rho(A)} |p(\lambda_i)| \tag{3.16}$$

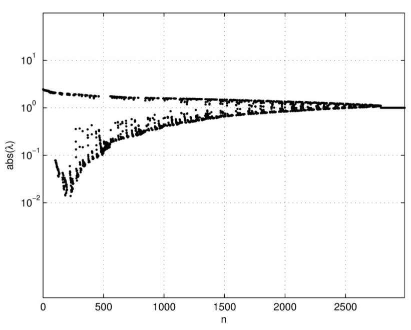
where P_m is the set of polynomials of degree less than or equal to m, $\rho(A)$ is the spectrum of A and $\kappa(\mathbf{X}) = ||\mathbf{X}||\,||\mathbf{X}^{-1}||$ is the condition number of the eigenvector matrix $\mathbf{X}$. Now observe that the application of a preconditioner to the system matrix $\mathbf{K}$ brings its spectrum closer to the unit circle. That is, there appears to be a single cluster of eigenvalues that lies in the vicinity of that circle. Therefore, if $p \in P_m$ has

Figure 3.5 Eigenvalue Modulus: $\sigma = 50.0$, $\gamma = 0.64$

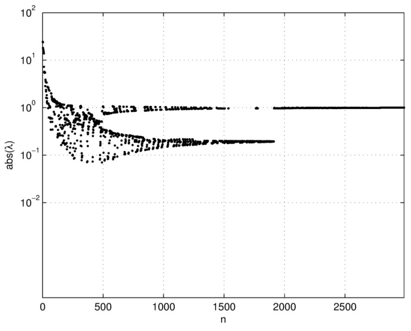
(a) No Preconditioner



(b) Diagonal Preconditioner

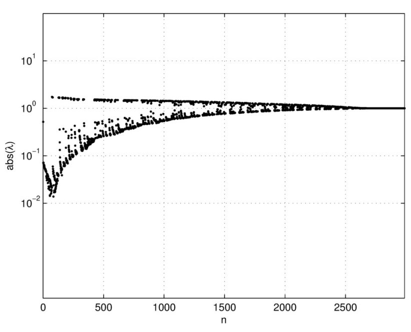


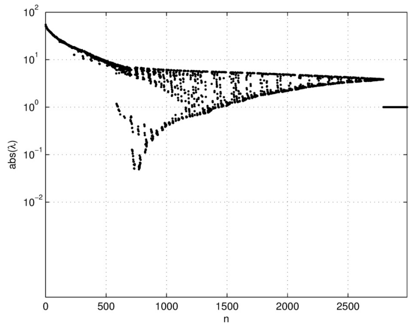
(c) ZFL Preconditioner



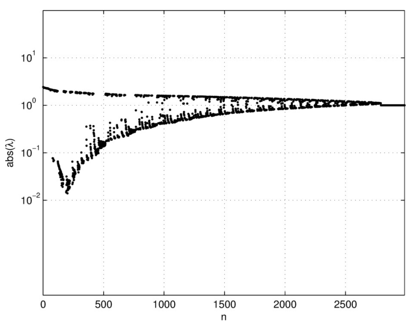
(d) Approximate Inverse

Preconditioner

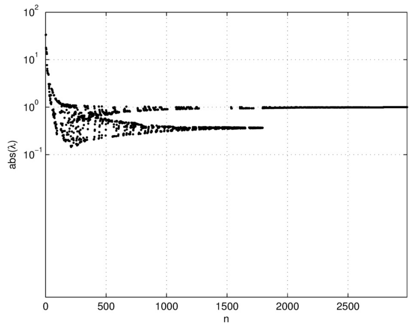




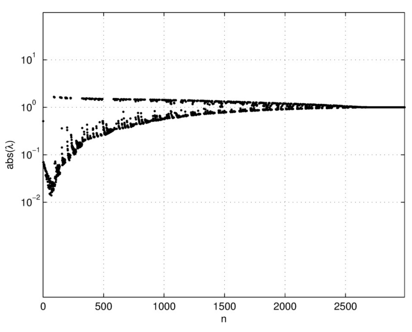
(a) No Preconditioner



(b) Diagonal Preconditioner

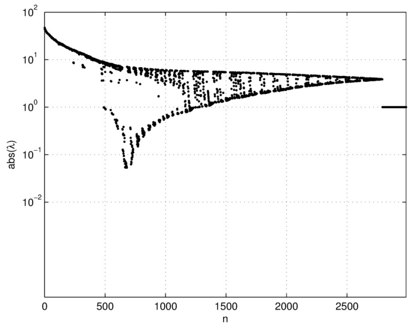


(c) ZFL Preconditioner

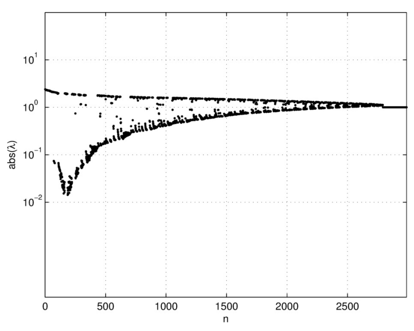


(d) Approximate Inverse Preconditioner

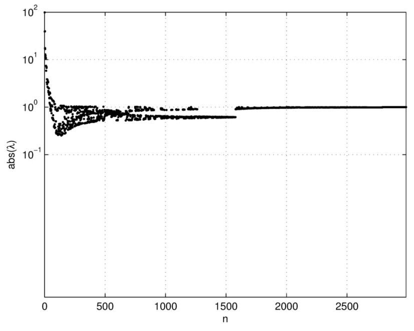
Figure 3.6 Eigenvalue Modulus: $\sigma = 50.0$, $\gamma = 1.28$



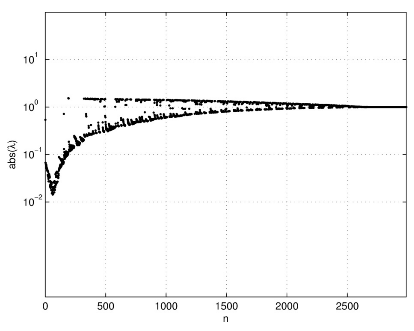
(a) No Preconditioner



(b) Diagonal Preconditioner



(c) ZFL Preconditioner



(d) Approximate Inverse Preconditioner

Figure 3.7 Eigenvalue Modulus: $\sigma = 50.0$, $\gamma = 2.56$

a zero in that cluster, it must remain small through it. Provided that the condition number of the eigenvector matrix of the preconditioned matrix does not grow very large, we would expect the residual to decrease very rapidly.

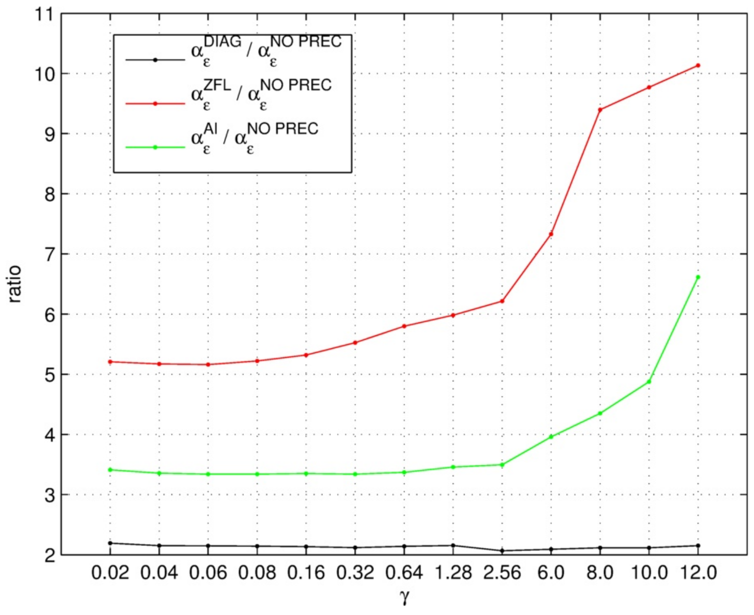
Good clustering of eigenvalues is therefore a sign of good convergence of GMRES. However, to quantify the possible improvements is very difficult in practice for it requires determination of the polynomial $p \in P_m$ that is minimized over the spectrum of the matrices in question. To our knowledge, there are no general results concerning the behavior of p with respect to a given spectrum. However, to gain some insight into the performance of the ZFL preconditioner, we have devised a simple *ratio indicator* in an attempt to quantify the relationship between the clustering of eigenvalues around unity and the convergence rate.

To see if there is a quantifiable relationship between this behavior and the convergence rate of GMRES, we compute the ratio of the number of eigenvalues located in a small neighborhood of the unit circle to the dimension of the system. That is, if the system size is N and there are N_ϵ eigenvalues in the small ϵ neighborhood of the unit circle, we define the ratio indicator α_ϵ to be:

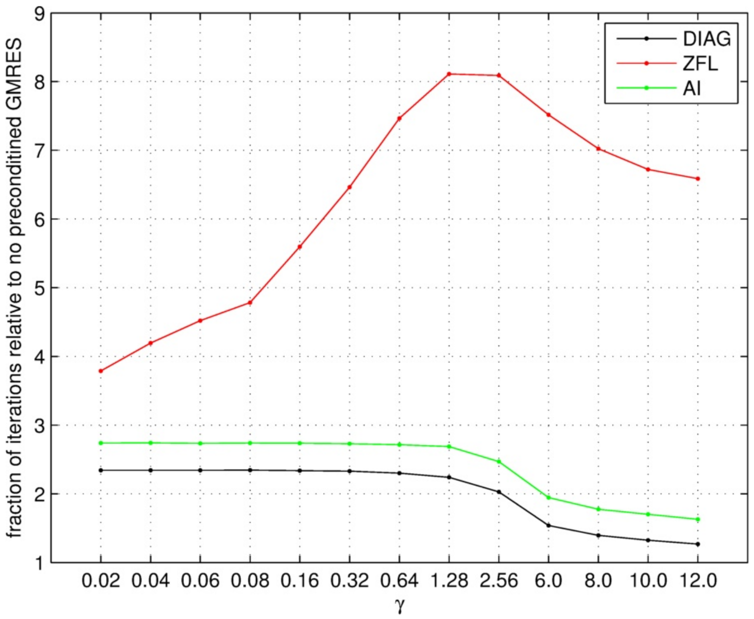
$$\alpha_\epsilon = \frac{N_\epsilon}{N} \tag{3.17}$$

To differentiate between different preconditioners, we shall superscript the above notation. Therefore, if we are considering the ratio indicator for the diagonal preconditioner we write: $\alpha_\epsilon^{\text{diag}}$. We compare these values with the number of iterations required to attain the specified residual tolerance in the Figure (3.8) where we superimpose the ratio indicators with the corresponding iteration plots of Figure (3.3).

These plots do indeed indicate a strong correspondence between the lowering of required number of iterations and clustering of eigenvalues around the unit circle. Especially in the case of the ZFL preconditioner, we observe that as the number

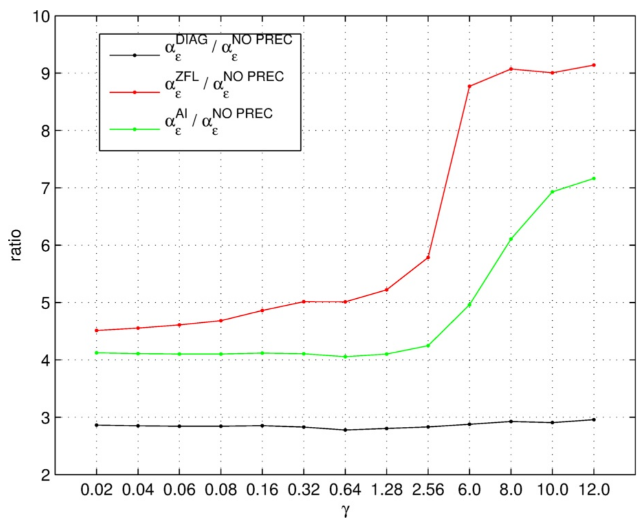


(a) ratio indicator vs. γ



(b) iteration count vs. γ

Figure 3.8 $\underline{\sigma = 50.0}$: Ratio Indicators and Iteration Counts, $\epsilon = 0.1$.

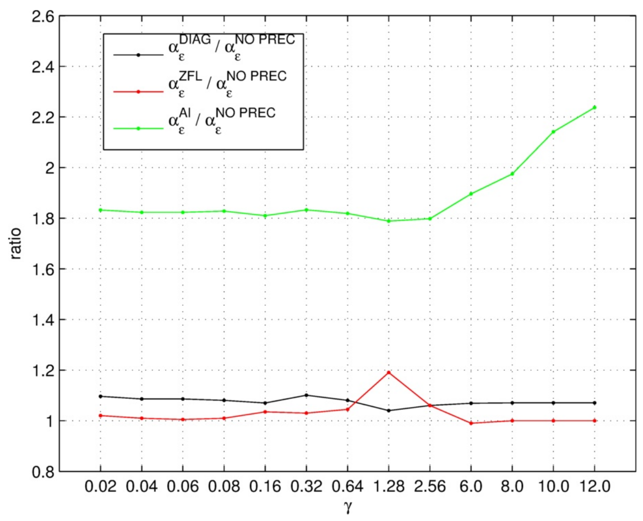


(a) ratio indicator vs. γ

Figure 3.9 $\underline{\sigma = 50.0}$: Ratio Indicators, $\epsilon = 0.2$.

of eigenvalues in the small neighborhood of the unit circle increases, the number of iterations decreases and that the gain is quite substantial. This behavior is consistent for values of $\gamma < 1.28$, and it holds true for the diagonal and approximate inverse preconditioners as well, which seem to be insensitive to the changes in γ in that parameter regime. The Figure (3.8) has been produced using the value of $\epsilon = 0.1$. For $\epsilon = 0.2$ and $\epsilon = 0.005$ the picture is quite different.

In all cases of ϵ considered, we observe that the performance of the diagonal preconditioner is not greatly affected by the changes in the parameter γ. Moreover, for values of $\gamma < 1.28$, at $\epsilon = 0.1$, the two ratios are very close. That is, the relative speedup obtained from diagonal preconditioning is proportional to the relative number of eigenvalues in the vicinity of the unit circle. Same could be said about the approximate inverse preconditioner as well, although the discrepancy in the two ratios is greater. For the ZFL preconditioner, at $\epsilon = 0.005$ we see a sharp change in $\alpha_\epsilon^{\text{zfl}}$ at $\gamma = 1.28$. Although the numerical values do not match, this curve describes the behavior of the corresponding iterations as a function of γ more closely since it captures the gain ($\gamma < 1.28$) and then the loss ($\gamma > 1.28$) in the relative number



(a) ratio indicator vs. γ

Figure 3.10 $\underline{\sigma = 50.0}$: Ratio Indicators, $\epsilon = 0.005$.

of iterations required. In any case, the figures do show that the value of $\gamma \approx 1.28$ is special and the reasons for that should be investigated further. Considering the number of parameters involved in the problem, having singled out a particular value of γ is a great simplification for future research. Thus far, we conclude that the application of the ZFL preconditioner results in the lowest GMRES iteration count since it clusters the eigenvalues of the matrix closer to the unit circle than the other preconditioners. Moreover, the ZFL preconditioner is very easy to construct. Given an assembly procedure for generating the FEM matrix $\mathbf{K}$, the preconditioner $\mathbf{P}_{\text{zfl}}$ can be obtained by simply setting $k = 0$ which results in a sparse matrix. Comparing with the computational effort required to find $\mathbf{P}_{\text{aip}}$, which in practice we have found to be a full matrix, the ZFL preconditioner is extremely cheap. Future work ought to aim at a more rigorous investigation of the relationship between a distribution of the spectrum of a ZFL preconditioned systems and the resulting GMRES iteration counts both as a function of γ as well as the wavenumber k.

CHAPTER 4

DOMAIN DECOMPOSITION METHOD FOR THE PERFECTLY MATCHED LAYER PROBLEM

4.1 Motivation

As noted at the end of Section (3.1), a domain decomposition method is a preconditioned iterative method in disguise. Indeed, many domain decomposition techniques can be characterized as preconditioned methods for the Schur complement system [74]. Since the PML problem is difficult to solve using Krylov subspace iterations [37], an investigation of an decomposition algorithm is therefore very natural. Moreover, as we explained in Chapter 2, domain decomposition algorithms are parallelizable in a straightforward manner, and thus allow considerations of very large problems. Since the layer itself can be partitioned into regions containing small number of unknowns, the additional potential benefit is that the PML region need not necessarily be made as small as possible to achieve computational efficiency. We begin the presentation of the method by recalling the 2D model problem of acoustic scattering and its truncation by the perfectly matched layer.

4.2 Model Problem and the PML Equations

Consider again the problem (1.19) whose geometry is depicted in in Figure (2.1). We recall, that the application of the PML technique to the numerical solution of that problem involves enclosing the scatterer by a thin layer as shown below in Figure (4.1).

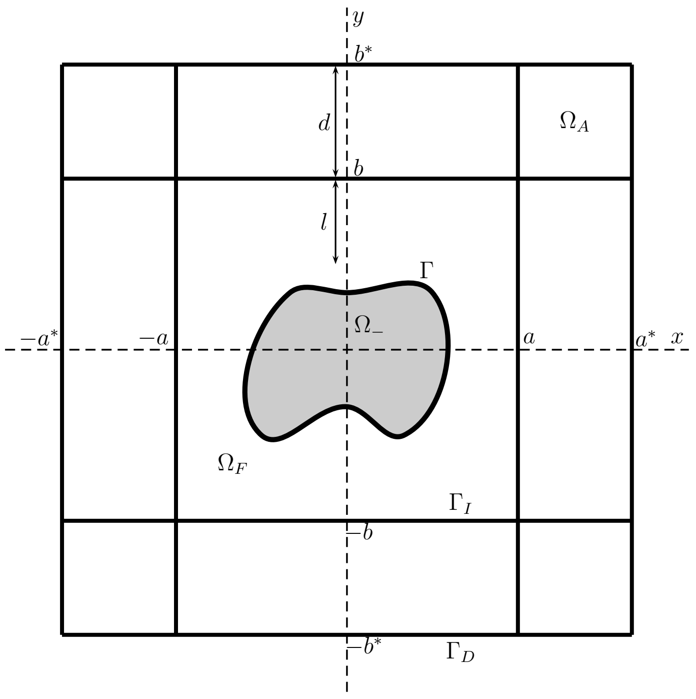


Figure 4.1 Problem domain truncated with PML.

The computational region obtained in the process is denoted by Ω_F. The PML region is denoted by Ω_A and it occupies the region of space between the PML interface Γ_I and domain boundary Γ_D. Its thickness is denoted by d and the minimum distance from the obstacle by l. Derivation of PML equations is now well known [48, section 3.3.4]. We follow the notation introduced in the previous chapter and denote by u_F^s and u_A^s the amplitudes of the scattered pressure waves in the physical and in the PML regions, respectively. These two coupled quantities satisfy the standard PML system of equations (see Section 3.2 for details):

$$\begin{cases} \Delta u_F^s + k^2 u_F^s = 0 \quad \text{in} \quad \Omega_F \\ \gamma_x^{-1}\partial_x\left(\gamma_x^{-1}\partial_x u_A^s\right) + \gamma_y^{-1}\partial_y\left(\gamma_y^{-1}\partial_y u_A^s\right) + k^2 u_A^s = 0 \quad \text{in} \quad \Omega_A \\ \partial_\nu u_F^s = -\partial_\nu u^i \quad \text{on} \quad \Gamma \\ u_F^s = u_A^s \quad \text{on} \quad \Gamma_I \\ \partial_\nu u_F^s = -\partial_\nu u_A^s \quad \text{on} \quad \Gamma_I \\ u_A^s = 0 \quad \text{on} \quad \Gamma_D \end{cases} \tag{4.1}$$

where:

$$\begin{aligned}\gamma_x(x) &= \begin{cases} 1 & x < a, \\ 1 + \frac{i}{k}\sigma_x(|x|) & a \leq |x| < a^* \end{cases} \\ \gamma_y(x) &= \begin{cases} 1 & y < b, \\ 1 + \frac{i}{k}\sigma_y(|y|) & b \leq |y| < b^* \end{cases}\end{aligned} \tag{4.2}$$

where ν is the vector normal to Γ_I pointing away from Ω_F.

4.3 DDM Formulation of the PML Problem

Analogously to the developments of section (4.5.1) we consider a partition of the computational domain $\Omega = \Omega_F \cup \Omega_A$ into N^d subdomains Ω_i, $i = 0, 1, \ldots, N^d - 1$, such that Ω_i is fully contained in either Ω_F and Ω_A and

- $\overline{\Omega} = \bigcup_{i=0}^{Nd-1} \overline{\Omega}_i$
- $\Omega_i \cap \Omega_j = \emptyset$ if $i \neq j$
- $\partial\Omega_i \cap \partial\Omega_j = \overline{\Sigma}_{ij} = \overline{\Sigma}_{ji}$ is the artificial interface separating Ω_i from Ω_j.

The requirement that either $\Omega_i \subset \Omega_F$ or $\Omega_i \subset \Omega_A$ is necessary to guarantee that the artificial interfaces align with the PML interface Γ_I. An example of an admissible decomposition is shown in Figure (4.2) below.

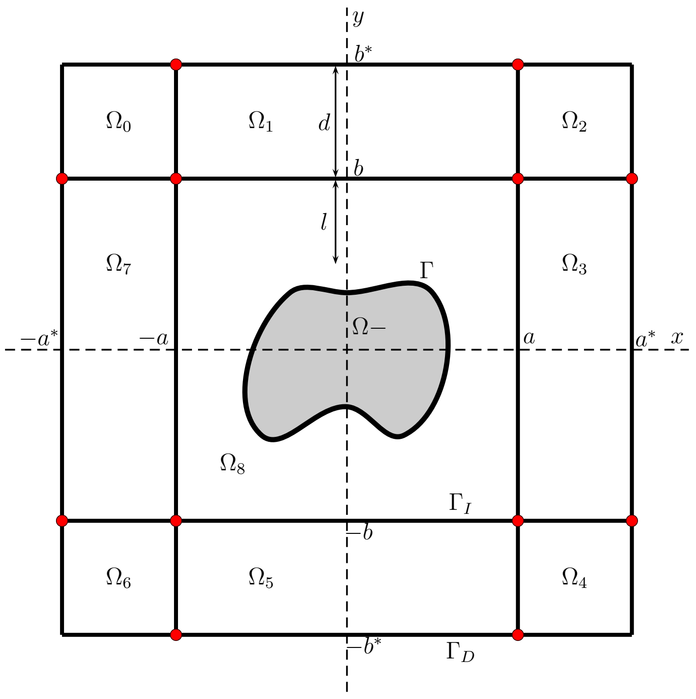


Figure 4.2 Admissible decomposition with cross-points (red dots).

Here, for convenience the subdomains $\Omega_i : i = 0, 1, \ldots, 8$ cover regions of the computational domain where the definition of the coefficient functions (3.2) remains unchanged. We note that while decompositions without cross-points are possible (for example, Ω_F and Ω_A constitute two subdomains), the most natural partitioning of the domain does introduce them. For that reason, when nodal finite element is used, the cross-point technique developed by Boubendir et al. [23] needs to be utilized. The next section shows how to adapt it to the PML problem (3.1). Before that presentation however, we formulate domain decomposition method for problem (3.1) under the assumption of absence of cross-points.

To that end, denote by Λ_F and Λ_A the sets of integer indices such that $\Lambda_F \cup \Lambda_A = \{0, 1, \ldots, N^d - 1\}$ with the property:

$$\Omega_F = \bigcup_{i \in \Lambda_F} \Omega_i, \quad \Omega_A = \bigcup_{i \in \Lambda_A} \Omega_i$$

We also define σ_i to be a set of integer indices j such that if $j \in \sigma_i$ then subdomain Ω_i shares an interface Σ_{ij} with subdomain Ω_j. Furthermore, we introduce local function spaces $X_i = \{v \in H^1(\Omega_i) : v|_{\Gamma_D} = 0\}$. In the domain decomposition formulation of the problem (3.1), the $(n+1)^{\text{th}}$ iteration step involves determination of the solution $u_i^{(n+1)} \in X_i$ to the local problem:

$$\begin{cases} \Delta u_i^{(n+1)} + k^2 u_i^{(n+1)} = 0 \quad \text{in} \quad \Omega_i \\ \partial_{\nu_i} u_i^{(n+1)} = -\partial_{\nu_i} u^i \quad \text{on} \quad \Gamma_i \\ \partial_{\nu_i} u_i^{(n+1)} + \Lambda\, u_i^{(n+1)} = g_{ij}^{(n)} \quad \text{on} \quad \Sigma_{ij} : j \in \sigma_i \end{cases} \tag{4.3}$$

if $i \in \Lambda_F$ or:

$$\begin{cases} \gamma_x^{-1}\partial_x \left(\gamma_x^{-1}\partial_x u_i^{(n+1)}\right) + \gamma_y^{-1}\partial_y \left(\gamma_y^{-1}\partial_y u_i^{(n+1)}\right) + k^2 u_i^{(n+1)} = 0 \quad \text{in} \quad \Omega_i \\ u_i^{(n+1)} = 0 \quad \text{on} \quad \Gamma_D \\ \partial_{\nu_i} u_i^{(n+1)} + \Lambda\, u_i^{(n+1)} = g_{ij}^{(n)} \quad \text{on} \quad \Sigma_{ij} : j \in \sigma_i \end{cases} \tag{4.4}$$

if $i \in \Lambda_A$. The function $g_{ij}^{(n)}$ is defined as:

$$g_{ij}^{(n)} = -\partial_{\nu_i} u_j^{(n)} + \Lambda u_j^{(n)} \tag{4.5}$$

on Σ_{ij}.

4.4 Convergence Analysis of DDM for PML Truncated Half-Plane Problem

To gain some insight into the behavior of the the iterative method defined in the previous section, we consider an exterior Helmholtz problem in the right-half plane:

$$\begin{cases} \Delta u + k^2 u = 0 \quad \text{for} \quad x > 0 \\ u(0,y) = -u^i(y) \\ \lim_{r\to+\infty} \sqrt{r}\left(\frac{\partial u}{\partial r} - iku\right) = 0 \end{cases} \tag{4.6}$$

where u^i is the prescribed incident field and $r = \sqrt{x^2+y^2}$.

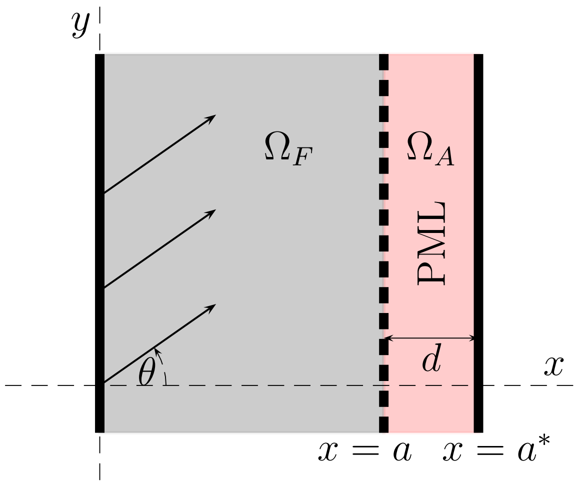


Figure 4.3 Half-plane problem geometry truncated with PML.

We introduce a PML of width d in the vertical strip $a < x < a^*$ to truncate the unbounded domain in the x-direction (Figure 3.1). Let us denote this region of space by Ω_A. The physical domain Ω_F is the strip $0 < x < a$ in which we wish to compute the solution. Let $u_A = u|_{\Omega_A}$ and $u_F =|_{\Omega_F}$. The PML equations are:

$$\begin{cases} \Delta u_F + k^2 u_F = 0 \quad \text{for} \quad 0 < x < a \\ \gamma^{-1}\partial_x\left(\gamma^{-1}\partial_x u_A\right) + \partial_y^2 u_A + k^2 u_A = 0 \quad \text{for} \quad a < x < a^* \\ u_F(0,y) = f(y) \\ u_F(a,y) = u_A(a,y) \\ \partial_x u_F(a,y) = \gamma^{-1}(a)\,\partial_x u_A(a,y) \\ u_A(a^*,y) = 0 \end{cases} \tag{4.7}$$

where:

$$\gamma(x) = \begin{cases} 1 & 0 < x < a \\ 1 + \frac{i\sigma^*}{k}(x-a)^p & a \leq x < a^* \end{cases} \tag{4.8}$$

This is a standard PML system of equations with the absorbing function $\sigma(x) = \sigma^*(x-a)^p$ where σ^* is a constant and p a positive integer. In this system, u_A and u_F are coupled through the continuity condition at $x = a$. Since $\gamma(a) = 1$, this condition is simply:

$$\begin{aligned} u_F &= u_A, \\ \partial_x u_A &= \partial_x u_F \end{aligned} \tag{4.9}$$

at $x = a$. Let us now assume that Λ is a transmission operator. The continuity equations are equivalent to:

$$\begin{aligned} \partial_x u_F + \Lambda u_F &= \partial_x u_A + \Lambda u_A, \\ -\partial_x u_A + \Lambda u_A &= -\partial_x u_F + \Lambda u_F. \end{aligned} \tag{4.10}$$

Using these equations we can decouple u_F and u_A and solve for them iteratively as follows. Introduce the iteration index n and define following local problems:

$$\begin{cases} \Delta u_F^{(n+1)} + k^2 u_F^{(n+1)} = 0 \quad \text{for} \quad 0 < x < a \\ u_F^{(n+1)}(0,y) = -u^i(y) \\ \partial_x u_F^{(n+1)}(a,y) + \Lambda u_F^{(n+1)}(a,y) = g_F^{(n)}(y) \end{cases} \tag{4.11}$$

and:

$$\begin{cases} \gamma^{-1}\partial_x\left(\gamma^{-1} u_A^{(n+1)}\right) + \partial_y^2 u_A^{(n+1)} + k^2 u_A^{(n+1)} = 0 \quad \text{for} \quad a < x < a^* \\ u_A^{(n+1)}(a^*,y) = 0 \\ -\partial_x u_A^{(n+1)}(a,y) + \Lambda u_A^{(n+1)}(a,y) = g_A^{(n)}(y) \end{cases} \tag{4.12}$$

where:

$$g_F^{(n)} = \partial_x u_A^{(n)} + \Lambda u_A^{(n)}, \quad \text{and} \quad g_A^{(n)} = -\partial_x u_F^{(n)} + \Lambda u_F^{(n)}. \tag{4.13}$$

The iteration is updated accordingly to the formulas derived in the earlier section:

$$g_F^{(n+1)} = -g_A^{(n)} + 2\Lambda u_A^{(n+1)}, \quad \text{and} \quad g_A^{(n+1)} = -g_F^{(n)} + 2\Lambda u_F^{(n+1)}. \tag{4.14}$$

These two sets of equations are analogues of Equation (4.5).

The convergence properties of the iterative scheme can be determined from the corresponding iteration operator. In the remained of this section we derive this operator using Fourier analysis arguments. The analysis of this section is a modification of an arguments utilized by Bermudéz et al. in [18] to study the accuracy of the PML method.

We begin by observing that the equations we are dealing with are linear. Therefore, to simplify the analysis we subtract off the true solutions and investigate instead whether or not the error converges to zero. More precisely, with u_F and u_A being the true solutions to (4.7), we define the error at the n^{th} iteration step as:

$$e_F^{(n+1)} = u_F^{(n+1)} - u_F \qquad \text{and} \qquad e_A^{(n)} = u_A^{(n)} - u_A. \tag{4.15}$$

The right hand side of the transmission conditions must be modified accordingly. Let us denote by $h_F^{(n)}$ the quantity:

$$h_F^{(n)} = g_F^{(n)} - \left.(\partial_x u_F + \Lambda u_F)\right|_{x=a}.$$

Similarly, for the problem in Ω_A we define $h_A^{(n)}$ to be:

$$h_F^{(n)}(y) = g_F^{(n)}(y) + \left.(\partial_x u_A - \Lambda u_A)\right|_{x=a}.$$

The equations we will therefore analyze are:

$$\begin{cases} \Delta e_F^{(n+1)} + k^2 e_F^{(n+1)} = 0 \quad \text{for} \quad 0 < x < a \\ e_F^{(n+1)}(0,y) = 0 \\ \partial_x e_F^{(n+1)}(a,y) + \Lambda e_F^{(n+1)}(a,y) = h_F^{(n)}(y) \end{cases} \tag{4.16}$$

and:

$$\begin{cases} \gamma^{-1}\partial_x\left(\gamma^{-1}\partial_x e_A^{(n+1)}\right) + \partial_y^2 e_A^{(n+1)} + k^2 e_A^{(n+1)} = 0 \quad \text{for} \quad a < x < a^* \\ e_A^{(n+1)}(a^*, y) = 0 \\ -\partial_x e_A^{(n+1)}(a, y) + \Lambda e_A^{(n+1)}(a, y) = h_A^{(n)}(y) \end{cases} \tag{4.17}$$

Recalling the continuity conditions at the PML interface, we immediately obtain:

$$h_F^{(n)} = \partial_x e_A^{(n)} + \Lambda e_A^{(n)}, \tag{4.18}$$

$$h_A^{(n)} = -\partial_x e_F^{(n)} + \Lambda e_F^{(n)} \tag{4.19}$$

while the update equations become:

$$h_F^{(n+1)} = -h_A^{(n)} + 2\Lambda e_A^{(n+1)}, \tag{4.20}$$

$$h_A^{(n+1)} = -h_F^{(n)} + 2\Lambda e_F^{(n+1)}. \tag{4.21}$$

The above equations for h are of course symmetric to those satisfied by functions g.

Now introduce the so called stretched coordinate:

$$\xi(x) = \int_a^x \gamma(s)\, ds = x + \frac{i\sigma^*}{k}\int_a^x (s-a)^p\, ds.$$

Chain rule gives:

$$\frac{\partial}{\partial x} = \frac{d\xi}{dx}\frac{\partial}{\partial \xi},$$

or equivalently:

$$\frac{\partial}{\partial \xi} = \frac{1}{\gamma}\frac{\partial}{\partial x}.$$

In the stretched coordinate, the PML problem (4.17) becomes:

$$\begin{cases} \partial_\xi^2 E_A^{(n+1)} + \partial_y^2 E_A^{(n+1)} + k^2 E_A^{(n+1)} = 0 \quad \text{for} \quad \xi(a) < \xi < \xi(a^*) \\ E_A^{(n+1)}(\xi(a^*), y) = 0 \\ -\partial_\xi E^{(n+1)}(\xi(a), y) + \Lambda E_A^{(n+1)}(\xi(a), y) = h_A^{(n)}(y) \end{cases} \tag{4.22}$$

where $e_A^{(n+1)}(x,y) = E_A^{(n+1)}(\xi(x),y)$.

We are now ready to apply the Fourier transform argument. The Fourier transform pair of a function $f(y)$ is:

$$\hat{f}(\omega) = \mathcal{F}\{f\} := \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} f(y)e^{-i\omega y}dy, \quad f(y) = \mathcal{F}^{-1}\left\{\hat{f}\right\} := \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} \hat{f}(\omega)e^{i\omega y}d\omega.$$

Taking the Fourier transform of equations (4.16) and (4.22) in y we get:

$$\begin{cases} \partial_x^2 \hat{e}_F^{(n+1)} + (k^2-\omega^2)\hat{e}_F^{(n+1)} = 0 \quad \text{for} \quad 0 < x < a \\ \hat{e}_F^{(n+1)}(0,\omega) = 0 \\ \partial_x \hat{e}_F^{(n+1)}(a,\omega) + \sigma(\omega)\hat{e}_F^{(n+1)}(a,\omega) = \hat{h}_F^{(n)}(\omega) \end{cases} \tag{4.23}$$

and

$$\begin{cases} \partial_\xi^2 \hat{E}_A^{(n+1)} + (k^2-\omega^2)\hat{E}_A^{(n+1)} = 0 \quad \text{for} \quad \xi(a) < \xi < \xi(a^*) \\ \hat{E}_A^{(n+1)}(\xi(a^*),\omega) = 0 \\ -\partial_\xi \hat{E}_A^{(n+1)}(\xi(a),\omega) + \sigma(\omega)\hat{E}_A^{(n+1)}(\xi(a),\omega) = \hat{h}_A^{(n)}(\omega) \end{cases} \tag{4.24}$$

where σ is the symbol of the iteration operator, that is:

$$\Lambda e(a,y) = \mathcal{F}^{-1}\left\{\sigma(\omega)\hat{e}(a,\omega)\right\}. \tag{4.25}$$

Now define:

$$\lambda(\omega) = i\sqrt{k^2-\omega^2}. \tag{4.26}$$

We immediately see that the solution to (4.23) is:

$$\hat{e}_F^{(n+1)}(x,\omega) = R_F^{(n+1)}(\omega)\, e^{\lambda(\omega)x} + L_F^{(n+1)}(\omega)\, e^{-\lambda(\omega)x}. \tag{4.27}$$

The boundary condition at $x = 0$ gives:

$$L_F^{(n+1)} = -R_F^{(n+1)}. \tag{4.28}$$

From the transmission condition at $x = a$ we get:

$$R_F^{(n+1)}\,(\lambda + \sigma)\,e^{\lambda\,a} + L_F^{(n+1)}\,(\sigma - \lambda)\,e^{-\lambda\,a} = \hat{h}_F^{(n)}. \tag{4.29}$$

Combining the two equations we find that:

$$R_F^{(n+1)} = \frac{\hat{h}_F^{(n)}}{(\lambda + \sigma)e^{\lambda\,a} + (\lambda - \sigma)e^{-\lambda\,a}}. \tag{4.30}$$

Similarly, the solution to equation (4.24) is of the form:

$$\hat{E}_F^{(n+1)}(\xi,\omega) = R_A^{(n+1)}(\omega)\,e^{\lambda(\omega)\xi} + L_A^{(n+1)}(\omega)\,e^{-\lambda(\omega)\xi},$$

or in the original variable:

$$\hat{e}_A^{(n+1)}(x,\omega) = R_A^{(n+1)}(\omega)\,e^{\lambda(\omega)\xi(x)} + L_A^{(n+1)}(\omega)\,e^{-\lambda(\omega)\xi(x)}. \tag{4.31}$$

The homogeneous Dirichlet boundary condition at $x = a^*$ gives:

$$L_A^{(n+1)} = -R_F^{(n+1)}\,e^{2\lambda\,\xi(a^*)} \tag{4.32}$$

whereas the transmission condition at $x = a$ yields:

$$R_A^{(n+1)}(\sigma - \lambda)e^{\lambda\,a} + L_A^{(n+1)}(\sigma + \lambda)e^{-\lambda\,a} = \hat{h}_A^{(n)}. \tag{4.33}$$

Combining the two equations we find that:

$$R_A^{(n+1)} = \frac{\hat{h}_A^{(n)}}{(\sigma - \lambda)e^{\lambda\,a} - (\sigma + \lambda)e^{2\lambda\,\xi(a^*)}e^{-\lambda\,a}}. \tag{4.34}$$

The Fourier modes of the interface unknowns $\hat{h}_F$ and $\hat{h}_A$ are updated according to the same formula as in Equation (4.14). That is:

$$\begin{aligned}\hat{h}_F^{(n+1)}(\omega) &= -\hat{h}_F^{(n)}(\omega) + 2\sigma(\omega)\hat{e}_F^{(n+1)}(\omega),\\ \hat{h}_A^{(n+1)}(\omega) &= -\hat{h}_A^{(n)}(\omega) + 2\sigma(\omega)\hat{e}_A^{(n+1)}(\omega).\end{aligned} \tag{4.35}$$

The update equations can be conveniently written in the matrix form:

$$\hat{\mathbf{h}}^{(n+1)}(\omega) = \hat{\mathbf{A}}\left(\omega;\sigma,k\right)\hat{\mathbf{h}}^{(n)}(\omega) \tag{4.36}$$

where $\hat{\mathbf{h}}^{(n)}\left(\omega\right) = \left(\hat{h}_F^{(n)}(\omega), \hat{h}_A^{(n)}(\omega)\right)^T$ and the matrix $\hat{\mathbf{A}}\left(\omega;\sigma,k\right)$ is the Fourier component in the modal decomposition of the iteration operator $\mathbf{A}\left(\omega;\Lambda,k\right)$ whose action on the vector $\mathbf{h}^{(n)}(y) = \left(h_F^{(n)}(y), h_F^{(n)}(y)\right)^T$ is defined by:

$$\mathbf{h}^{(n+1)}(y) = \mathbf{A}\left(y;\Lambda,k\right)\mathbf{h}^{(n)}(y). \tag{4.37}$$

The action of the operator $\mathbf{A}\left(y;\Lambda,k\right)$ is expressed in terms of its Fourier decomposition:

$$\begin{aligned}\mathbf{A}\left(y;\Lambda,k\right)\,\mathbf{h}^{(n)}(y) &= \mathbf{h}^{(n+1)}(y) && (4.38)\\ &= \mathcal{F}^{-1}\left\{\hat{\mathbf{h}}^{(n+1)}(\omega)\right\}(y) && (4.39)\\ &= \mathcal{F}^{-1}\left\{\hat{\mathbf{A}}\left(\omega;\sigma,k\right)\hat{\mathbf{h}}^{(n)}(\omega)\right\}(y). && (4.40)\end{aligned}$$

To find the entries of $\hat{\mathbf{A}}\left(\omega;\sigma,k\right)$ we substitute for $\hat{e}_F^{(n+1)}(\omega)$ and $\hat{e}_A^{(n+1)}(\omega)$ in Equations (4.35) which gives:

$$\hat{\mathbf{A}}\left(\omega;\sigma,k\right) = \begin{pmatrix} 0 & -1 + 2\sigma\frac{e^{\lambda a} - e^{2\lambda\xi(a^*)}e^{-\lambda a}}{(\sigma-\lambda)e^{\lambda a} - (\sigma+\lambda)e^{2\lambda\,\xi(a^*)}e^{-\lambda a}} \\ -1 + 2\sigma\frac{e^{\lambda a} - e^{-\lambda a}}{(\lambda+\sigma)e^{\lambda a} + (\lambda-\sigma)e^{-\lambda a}} & 0 \end{pmatrix} \tag{4.41}$$

For convenience, we rewrite the above matrix as:

$$\hat{\mathbf{A}}\left(\omega;\sigma,k\right) = \begin{pmatrix} 0 & A(\omega;\sigma,k) \\ B(\omega;\sigma,k) & 0 \end{pmatrix} \tag{4.42}$$

Proposition 4.4.1. The convergence factor μ of the domain decomposition method is:

$$\mu = \max_{\omega\in\mathbb{R}} |A(\omega)B(\omega)| = \max_{\omega\in\mathbb{R}} |\rho^2(\hat{\mathbf{A}}\left(\omega;\sigma,k\right))|$$

where ρ denotes the spectral radius.

Proof. For simplicity we will look at the even number of iterations. Using equation (4.42) we find that equation (4.36) in component form is:

$$\begin{aligned} \hat{h}_F^{(2n)}(\omega) &= A(\omega;\sigma,k)\hat{h}_A^{(2n-1)}(\omega) \\ \hat{h}_A^{(2n)}(\omega) &= B(\omega;\sigma,k)\hat{h}_F^{(2n-1)}(\omega) \end{aligned} \tag{4.43}$$

for $n \in \mathbb{Z}^+$. Expanding the $(2n-1)$ terms using the above formula yields following recurrence relation for $\hat{h}_F^{(2n)}(\omega)$ and $\hat{h}_A^{(2n)}(\omega)$:

$$\begin{aligned} \hat{h}_F^{(2n)}(\omega) &= A(\omega;\sigma,k)B(\omega;\sigma,k)\hat{h}_F^{(2n-2)}(\omega) \\ \hat{h}_A^{(2n)}(\omega) &= A(\omega;\sigma,k)B(\omega;\sigma,k)\hat{h}_A^{(2n-2)}(\omega) \end{aligned} \tag{4.44}$$

Upon defining $\mu = \max_{\omega\in\mathbb{R}} |A(\omega)B(\omega)|$ we can write:

$$\|\hat{\mathbf{h}}^{2n}\| \leq \mu \|\hat{\mathbf{h}}^{2n-2}\|$$

The proof now follows by considering the square of the $l2$-norm of the iterates:

$$\begin{aligned} \|\mathbf{h}^{(2n)}\|^2 &= \|\mathcal{F}^{-1}\left\{\hat{\mathbf{h}}^{(2n)}\right\}\|^2 \\ &\leq \|\mathcal{F}^{-1}\left\{\mu\,\hat{\mathbf{h}}^{(2n-2)}\right\}\|^2 \\ &\leq \mu\|\mathcal{F}^{-1}\left\{\hat{\mathbf{h}}^{(2n-2)}\right\}\|^2 \\ &= \mu\|\mathbf{h}^{(2n-2)}\|^2 \end{aligned}$$

The relationship between μ and the spectral radius ρ of the matrix $\hat{\mathbf{A}}(\omega;\sigma,k)$ is:

$$\sqrt{\mu} = \max_{\omega\in\mathbb{R}} |\sqrt{A(\omega)B(\omega)}| = \max_{\omega\in\mathbb{R}} \rho\left(\hat{\mathbf{A}}(\omega;\sigma,k)\right) \tag{4.45}$$

which is the rate of convergence factor over a single iteration. $\square$

Remark. The Jacobi type iteration for the system (4.16) - (4.17) converges if and only if:

$$\mu < 1. \tag{4.46}$$

Therefore, for convergence, we must choose the transmission operator Λ such that it satisfies the conclusions of the above remark. As the next theorem will show, the convergence is guaranteed when the square-root transmission operator is used.

Theorem 4.4.2. *The Jacobi type method converges if the symbol of the transmission operator Λ is:*

$$\sigma(\omega) = -i\sqrt{k^2 - \omega^2} = -\lambda(\omega) \tag{4.47}$$

Proof. The result is obtained upon substituting $\sigma = -\lambda$ into Equation (4.41). Algebraic simplification yields:

$$\hat{\mathbf{A}}\,(\omega; -\lambda, k) = \begin{pmatrix} 0 & e^{2\lambda(\xi(a^*)-a)} \\ -e^{2\lambda\, a} & 0 \end{pmatrix} \tag{4.48}$$

According to the Proposition (4.4.1) and the following remark, the Jacobi method converges provided that the modulus of the product of the two non zero entries is less than one, or equivalently, that the spectral radius of $\hat{\mathbf{A}}\,(\omega; -\lambda, k)$ is less than one. Consequently:

$$\begin{aligned} \rho(\hat{\mathbf{A}}\,(\omega; -\lambda, k)) &= \left| e^{\lambda(\omega)\xi(a^*)} \right| \\ &= \left| e^{\lambda(\omega)\left(a^* + \frac{i\sigma^* d^{p+1}}{k(p+1)}\right)} \right| \\ &= \begin{cases} e^{-\sqrt{k^2-\omega^2}\,\frac{\sigma^* d^{p+1}}{k(p+1)}} & |k| > |\omega| \\ e^{-\sqrt{\omega^2-k^2}\,a^*} & |k| < |\omega| \end{cases} \end{aligned} \tag{4.49}$$

which is less than one for all ω since the arguments of the exponential function remain negative. Thus, the convergence is indeed guaranteed over the whole spectrum. □

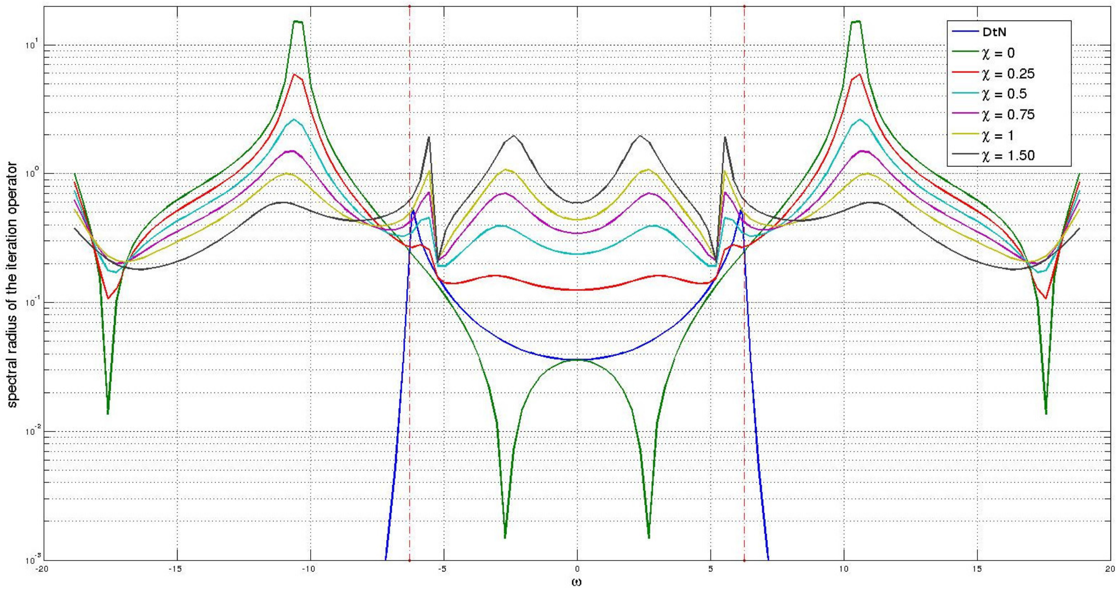


Figure 4.4 Spectral radius of the iteration operator.

The choice of the symbol σ leads to a non-local transmission operator due to the presence of a rational function in the inverse Fourier transform integral (see Equation 4.25). As explained in Chapter 2, from a computational perspective, this is generally expensive and difficult to implement. For that reason, to solve the 2D model PML problem using DDM we will localize the square-root transmission operator using the procedure presented in Section (2.3.1).

Since the transmission operator used in the EMDA [23] type DDM with a constant parameter χ is usually easier to implement, we compare the spectra of the corresponding iteration operators with that given by Equation (4.49) corresponding to the square-root operator. The Figure (4.4) below shows the spectral radii curves for various values of the parameter χ. The blue curve corresponds to the square-root transmission operator of Theorem (4.4.2). Figure (4.4) has been generated with $k = 2\pi$, $a = 1$, $d = 1$, $\sigma = 5$ and $p = 2$. For the EMDA transmission operator with $\sigma = -ik(1 + i\chi)$, we observe the dampening of the error in the propagating part of the spectrum and a slight amplification of the evanescent error components. The value $\chi = 1$ seems to be optimal in the sense that it brings the spectral radius of

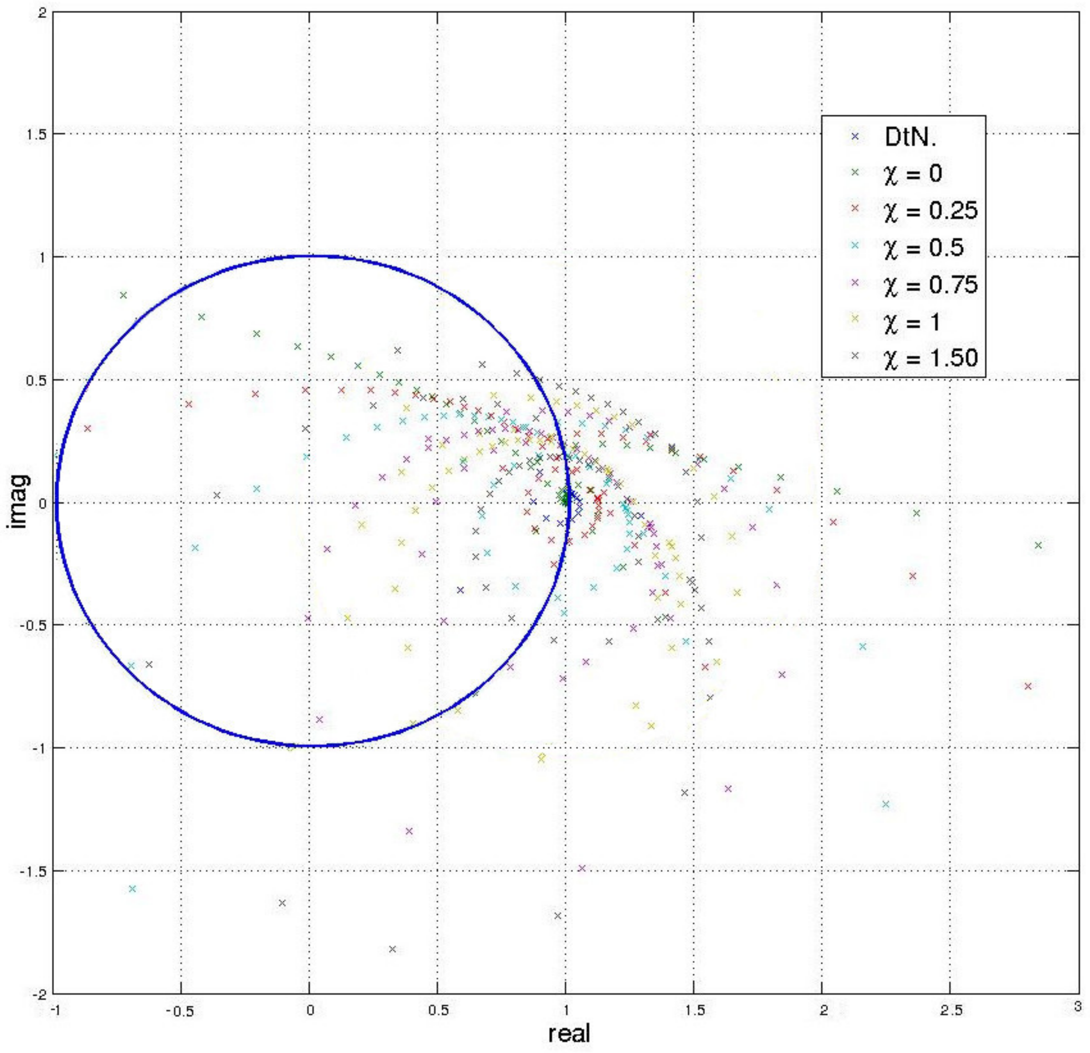


Figure 4.5 Sample of the continuous spectrum of the iteration operators.

the iteration operator under one over all modes considered in this test. In principle, an optimization problem could be posed for an optimal value of χ, but such a task is very difficult due to a complicated dependence of the spectral radius on the PML parameters in the general case. Fortunately, such a procedure is not necessary in practice, when a discrete problem on a bounded domain is solved using GMRES. The reason for this is that the spectrum of the iteration operator is now discrete and bounded. Moreover, the eigenvalues of the corresponding iteration equation cluster around one as the Figure (4.5) below shows.

We do observe clustering of eigenvalues around one which is desirable if we wish to solve the DDM iteration equation by a Krylov subspace method such as GMRES [72]. While the convergence of GMRES is guaranteed, the clustering of eigenvalues around the unit circle will make it very rapid [71, 75].

4.5 Finite Element Discretization

We are going to approximate the solution to the local problems using nodal finite element method. To that end let $u_i^h \in X_i^h$ be the finite element approximation to the solution $u_i^{(n+1)}$ of the problems (4.50) and (4.51). In the case of Λ being a constant, the finite element discretization of the local problems is straightforward. In fact, it is a simple modification of the problem (2.35) involving redefinition of functionals a_i^h and l_i^h to account for non-constant coefficients in the PML problem and homogeneous Dirichlet boundary condition on Γ_D. The situation is different when we localize the square-root operator using Padé series. In that case, the transmission conditions are given by Equations (2.32) and the local problems are coupled to the auxiliary surface functions by Equations (2.33). Consequently, the local PML problems (4.50) and (4.51) respectively, must be modified to:

$$\begin{cases} \Delta u_i^{(n+1)} + k^2 u_i^{(n+1)} = 0 \quad \text{in} \quad \Omega_i \\ \partial_{\nu_i} u_i^{(n+1)} = -\partial_{\nu_i} u^i \quad \text{on} \quad \Gamma_i \\ \partial_{\nu_i} u_i^{(n+1)} - ik\left(C_0 u_i^{(n+1)} + \sum_{l=1}^{N_p} A_l \nabla_{\Sigma_{ij}} \cdot \left(k_\epsilon^{-2} \nabla_{\Sigma_{ij}} \varphi_{i(l)}^{(n+1)}\right)\right) = g_{ij}^{(n)} \quad \text{on} \quad \Sigma_{ij} : j \in \sigma_i \\ \left(1 + B_l \nabla_{\Sigma_{ij}} \cdot \left(k_\epsilon^{-2} \nabla_{\Sigma_{ij}}\right)\right) \varphi_{i(l)}^{(n+1)} = u_{i(l)}^{(n+1)} \quad \Sigma_{ij} : j \in \sigma_i, \quad l = 1, \ldots, N_p \end{cases} \tag{4.50}$$

if $i \in \Lambda_F$ or:

$$\begin{cases} \gamma_x^{-1}\partial_x\left(\gamma_x^{-1}\partial_x u_i^{(n+1)}\right) + \gamma_y^{-1}\partial_y\left(\gamma_y^{-1}\partial_y u_i^{(n+1)}\right) + k^2 u_i^{(n+1)} = 0 \quad \text{in} \quad \Omega_i \\ u_i^{(n+1)} = 0 \quad \text{on} \quad \Gamma_D \\ \partial_{\nu_i} u_i^{(n+1)} - ik\left(C_0 u_i^{(n+1)} + \sum_{l=1}^{N_p} A_l \nabla_{\Sigma_{ij}} \cdot \left(k_\epsilon^{-2}\nabla_{\Sigma_{ij}}\varphi_{i(l)}^{(n+1)}\right)\right) = g_{ij}^{(n)} \quad \text{on} \quad \Sigma_{ij} : j \in \sigma_i \\ \left(1 + B_l \nabla_{\Sigma_{ij}} \cdot \left(k_\epsilon^{-2}\nabla_{\Sigma_{ij}}\right)\right)\varphi_{i(l)}^{(n+1)} = u_{i(l)}^{(n+1)} \quad \Sigma_{ij} : j \in \sigma_i, \quad l = 1, \ldots, N_p \end{cases} \tag{4.51}$$

where $g_{ij}^{(n)}$ and $g_{ij}^{(n+1)}$ are defined by Equations (2.33) and (2.34)

The variational formulation of the above system is analogous to that presented in Equations (2.37) and (2.38). We rewrite it here for convenience since we will refer to it in the next section:

$$\begin{aligned} a_i^h\left(u_i^h, v_i^h\right) - ikC_0 \sum_{j\in\sigma_i} c_{ij}^h\left(u_i^h, v_i^h\right) \\ + ik\sum_{j\in\sigma_i}\sum_{l=1}^{N_p} A_l\, c_{ij}^h\left(k_\epsilon^{-2}\nabla_{\Sigma_{ij}}\varphi_{i(l)}^h, \nabla_{\Sigma_{ij}} v_i^h\right) = l_i^h\left(v_i^h\right) + \sum_{j\in\sigma_i} c_{ij}^h\left(g_{ij}^h, v_i^h\right) \end{aligned} \tag{4.52}$$

$$c_{ij}^h\left(\varphi_{i(l)}^h, v_i^h\right) - B_l\, c_{ij}^h\left(k_\epsilon^{-2}\nabla_{\Sigma_{ij}}\varphi_{i(l)}^h, \nabla_{\Sigma_{ij}} v_i^h\right) = c_{ij}^h\left(u_i^h, v_i^h\right) \quad j \in \sigma_i, \quad l = 1, \ldots, N_p \tag{4.53}$$

Definitions of the functionals a_i^h and l_i^h depend on whether i belongs to Λ_F or Λ_A (see Section 4.3). More precisely:

$$\begin{aligned} a_i^h(u,v) &= \int_{\Omega_i^h} \nabla u \cdot \nabla v\, dA - k^2 \int_{\Omega_i^h} uv\, dA \\ c_{ij}(u,v) &= \int_{\Sigma_{ij}^h} u\, v\, ds \\ l_i^h(v) &= \int_{\Gamma_i^h} gv\, ds \end{aligned} \tag{4.54}$$

if $i \in \Lambda_F$ and:

$$\begin{aligned} a_i^h(u,v) &= \int_{\Omega_i^h} \gamma_y \gamma_x^{-1} \partial_x u \, \partial_x v \, dA + \int_{\Omega_i^h} \gamma_x \gamma_y^{-1} \partial_y u \, \partial_y v \, dA - k^2 \int_{\Omega_i^h} \gamma_x \gamma_y uv \, dA \\ c_{ij}(u,v) &= \int_{\Sigma_{ij}^h} u \, v \, ds \qquad (4.55) \\ l_i^h(v) &= 0 \end{aligned}$$

if $i \in \Lambda_A$.

The Figure (4.2) shows a natural decomposition of the PML problem domain. Red dots correspond to the vertices in the nodal finite element method which support three or more degrees of freedom. As the next section explains, these mesh points are problematic in the domain decomposition formulation of the problem when nodal finite element method is used. The discretization method as presented in this section is no longer applicable in that case. The algorithm that we discuss next, developed by Y. Boubendir et al. [23], enables formulation of non-overlapping methods over domain partitions that contain cross-points. We also show how that method can be adapted to handle the coupled variational problem (4.52) - (4.53) arising from the use of the Padé type transmission operator (2.27).

4.5.1 Decomposition with Cross-Points

We have seen that formulation of non-overlapping domain decomposition methods initiated relies on specification of appropriate matching conditions at subdomain interfaces. Derivation of the algorithms of Sections (2.2) and (4.3) made an extensive use of continuity conditions (2.4). However, this approach suffers from a characteristic difficulty which lies in the treatment of the points shared by more than two subdomains. These points are referred to as *cross-points.* In Figure (4.2) these special interface points are marked with red dots. Indeed, for discretizations based on a typical nodal finite element method, such points may support one or more degrees of freedom, each of them shared by more than two subdomains. The usual equivalent writing of the matching conditions at the subdomain interfaces, on which the Lions and the Després methods are based, does not remain valid at these points. In [15, 23], Boubendir and Bendali presented a stable and convergent reformulation of the original method which efficiently overcomes these difficulties. In essence, their approach consists of preservation of continuity of the finite element spaces at the cross-points.

To begin a brief presentation of the resulting algorithm we recall that X_i^h denotes the finite dimensional subspace of the solution space $X_i = \{v \in H^1(\Omega_i) : v|_{\Gamma_D} = 0\}$ consisting of piecewise linear polynomials. By u_i^h we denote the approximation to u_i in X_i^h. We first observe that any function $v_i^h \in X_i^h$ can be decomposed in the following manner:

$$v_i^h = v_{iI}^h + \sum_{j \in \sigma_i} v_{ij}^h + v_c^h \tag{4.56}$$

Definitions of functions v_{iI}^h, v_{ij}^h and v_c^h are motivated by the partitioning of the degrees of freedom of the finite element space X_i^h:

- all nodal values of v_{iI}^h are zero on the closure of any artificial interface Σ_{ij} separating Ω_i from a neighboring domain Ω_j for all $j \in \sigma_i$.
- all nodal values of v_{ij}^h are zero, except those located at the interior of the artificial interface Σ_{ij}. In particular, function v_{ij}^h vanish at cross-points.
- all nodal values of v_c^h are zero, except those corresponding to cross-points, defined here as points on the closure of Σ_{ij} but not lying in its interior. In this manner, points that are at the junctions of Σ_{ij} and the physical boundaries Σ and Γ, are also counted as cross-points.

The solution to the problem (3.1) belongs to the space $X = \{v \in H^1(\Omega) : v|_\Gamma = 0\}$. We shall denote by X^h a subspace of X consisting of piecewise linear polynomials.

We note that X_i^h are subspaces of X^h. Also, we observe that while v_{iI}^h and v_c^h are functions in X^h, functions v_{ij}^h are not. Let us now denote by X_c^h the subspace of X^h spanned by v_c^h, and by X_B^h the *broken* space spanned by functions of the form:

$$v^h = \sum_{i=1}^{N_d-1} \left(v_{iI}^h + \sum_{j\in\sigma_i} v_{ij} \right) + v_c^h \tag{4.57}$$

Note that X^h is a subspace of X_B^h containing exactly those members of X_B^h whose v_{ij}^h components are continuous functions in X^h. This continuity is expressed by a matching condition on the interfaces which modifies (4.52) - (4.53) into a system:

$$a_i^h \left(u_{iI}^h + \sum_{j\in\sigma_i} u_{ij}^h + v_c^h,\, v_{iI}^h \right) = l_i^h \left(v_{iI}^h \right) \quad \forall v_{iI}^h \in X_i^h \tag{4.58}$$

$$\begin{aligned} a_i^h \left(u_{iI}^h + \sum_{j\in\sigma_i} u_{ij}^h + v_c^h,\, v_{ij}^h \right) &- ikC_0 \sum_{j\in\sigma_i} c_{ij}^h \left(u_{ij}^h, v_{ij}^h \right) + \\ &+ ik \sum_{j\in\sigma_i} \sum_{l=1}^{N_p} A_l\, c_{ij}^h \left(k_\epsilon^{-2} \nabla_{\Sigma_{ij}} \varphi_{i(l)}^h, \nabla_{\Sigma_{ij}} v_{ij}^h \right) = \\ &= l_i^h \left(v_{ij}^h \right) + \sum_{j\in\sigma_i} c_{ij}^h \left(g_{ij}^h, v_{ij}^h \right) \quad \forall v_{ij} \in X_i^h, j \in \sigma_i \end{aligned} \tag{4.59}$$

$$c_{ij}^h \left(\varphi_{i(l)}^h, v_{ij}^h \right) - B_l\, c_{ij}^h \left(k_\epsilon^{-2} \nabla_{\Sigma_{ij}} \varphi_{i(l)}^h, \nabla_{\Sigma_{ij}} v_{ij}^h \right) = c_{ij}^h \left(u_{ij}^h, v_{ij}^h \right) \quad j \in \sigma_i, \quad l = 1, \ldots, N_p \tag{4.60}$$

subject to a coupling condition:

$$\sum_{i=1}^{N_d-1} a_i^h \left(u_{iI} + \sum_{j\in\sigma_i} u_{ij}^h + u_c^h,\, v_c^h \right) = \sum_{i=1}^{N_d-1} l_i^h \left(v_c^h \right) \quad \forall v_c^h \in X_c^h \tag{4.61}$$

Functionals a_i^h and l_i^h are defined in exactly the same way as in (4.54) and (4.55). However, the discrete functional c_{ij}^h used to express continuity of the solution on the interface in a variational form needs to be adjusted. To this end let us define $\tilde{\Sigma}_{ij}^h$ to be the part of Σ_{ij}^h obtained by excluding from it edges having a cross-point as one of its endpoints (see Figure 4.6 below).

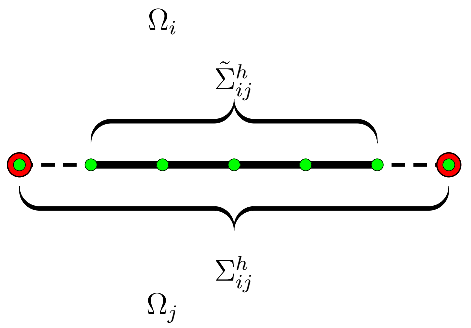


Figure 4.6 Interface decomposition, red dots - cross-points, green dots - mesh nodes.

Now we redefine c_{ij}^h to be:

$$c_{ij}^h\left(u_{ij}^h, v_{ij}^h\right) = \int_{\tilde{\Sigma}_{ij}^h} u_{ij}^h v_{ij}^h \, ds \tag{4.62}$$

The variational system (4.58) - (4.60) can be written in the following matrix form:

$$\begin{bmatrix} \mathbf{A}_{11} & & & & \mathbf{A}_{1c} \\ & \mathbf{A}_{22} & & & \mathbf{A}_{2c} \\ & & \ddots & & \vdots \\ & & & \mathbf{A}_{NN} & \mathbf{A}_{Nc} \\ \mathbf{A}_{c1} & \mathbf{A}_{c2} & \dots & \mathbf{A}_{cN} & \mathbf{A}_{cc} \end{bmatrix} \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \vdots \\ \mathbf{w}_N \\ \mathbf{w}_c \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ \vdots \\ \mathbf{f}_N \\ \mathbf{f}_c \end{bmatrix} \tag{4.63}$$

where $\mathbf{A}_{ii}$, $\mathbf{A}_{ic}$, $\mathbf{A}_{ci}$ for $i = 1, ..., N$ are matrices related to the discretization of the variational problem on Ω_i. Each $\mathbf{w}_i$ therefore, satisfies

$$\mathbf{A}_{ii}\mathbf{w}_i = \mathbf{f}_i - \mathbf{A}_{ic}\mathbf{w}_c \tag{4.64}$$

The above linear system can be solved once we obtain u_c which is a solution to

$$\left(\mathbf{A}_{cc} - \sum_{i=1}^{N} \mathbf{A}_{ci}(\mathbf{A}_{ii})^{-1}\mathbf{A}_{ic}\right)\mathbf{w}_c = \mathbf{f}_c - \sum_{i=i}^{N} \mathbf{A}_{ci}(\mathbf{A}_{ii})^{-1}\mathbf{f}_i \tag{4.65}$$

It should be noted that system (4.65) is small since it relates to the degrees of freedom supported by the cross points only. Furthermore, for effective computation, the iterative procedure should proceed as follows:

- Initialization:
  - perform an LU factorization of each matrix $\mathbf{A}_{ii}$ for $i = 1, ..., N$,
  - do a forward backward sweep to compute $(\mathbf{A}_{ii})^{-1}\mathbf{A}_{ic}$ for $i = 1, ..., N$,
  - form and carry out an LU factorization of $\left(\mathbf{A}_{cc} - \sum_{i=1}^{N} \mathbf{A}_{ci}(\mathbf{A}_{ii})^{-1}\mathbf{A}_{ic}\right)$,
- For each iteration:
  - do a forward backward sweep to compute $(\mathbf{A}_{ii})^{-1}\mathbf{f}_i$ for $i = 1, ..., N$,
  - form $\mathbf{f}_c - \sum_{i=i}^{N} \mathbf{A}_{ci}(\mathbf{A}_{ii})^{-1}\mathbf{f}_i$ and do a forward backward sweep to compute u_c,
  - obtain $\mathbf{w}_i$ for $i = 1, ..., N$ from (4.64) by means of basic linear algebra computations.

We observe that the main computational cost of each iteration can be attributed to the solution of two sparse linear systems $(\mathbf{A}_{ii})^{-1}\mathbf{f}_i$ and $\mathbf{w}_i = (\mathbf{A}_{ii})^{-1}(\mathbf{f}_i - \mathbf{A}_{ic}\mathbf{w}_c)$. However, performing the factorization of $\mathbf{A}_{ii}$ and consequently forming the matrix $\mathbf{A}_{ii}^{-1}\mathbf{A}_{ic}$ in the initialization phase of the algorithm essentially reduces the work per iteration to two back substitutions. Inversion of a dense system related to cross-points (4.65) needs to be performed at each iteration as well. Again, the bulk of the computational work necessary to complete this task can be performed in the initialization phase where the cross-point matrix is being formed and factorized. Furthermore, since the number of cross-points is small relative to the total number of degrees of freedom, the back substitution step that gives u_c can be performed very quickly, with essentially no penalty to the per iteration run time of the method.

Determination of Auxiliary Functions in Cross-Point DDM. The cross-point DDM needs to be modified in order to accommodate the localization of the transmission operator using Padé series as in Equation (2.27). Since the artificial interfaces Σ_{ij} are no longer closed, integration by parts produces non trivial boundary terms which are not present in the variational problem (4.58) - (4.60), thereby invalidating it. To see this, consider:

$$\begin{aligned} c_{ij}^h(\Lambda u_i^h, v_i^h) &= \int_{\Sigma_{ij}^h} \Lambda u_i^h v_i^h ds \\ &= -ikC_0 \int_{\Sigma_{ij}^h} u_i^h v_i^h ds - ik \sum_{l=1}^{N_p} A_l \int_{\Sigma_{ij}^h} \nabla_{\Sigma_{ij}^h} \cdot \left(k_\epsilon^{-2} \nabla_{\Sigma_{ij}^h} \varphi_{i(l)}^h \right) v_i^h ds \end{aligned} \tag{4.66}$$

Consequently, application of the integration by parts to the last integral yields:

$$\begin{aligned} \int_{\Sigma_{ij}^h} \nabla_{\Sigma_{ij}^h} \cdot \left(k_\epsilon^{-2} \nabla_{\Sigma_{ij}^h} \varphi_{i(l)}^h \right) v_i^h ds &= v_i^h \, k_\epsilon^{-2} \nabla_{\Sigma_{ij}^h} \varphi_{i(l)}^h \cdot \boldsymbol{\tau} \Big|_{\partial \Sigma_{ij}^h} \\ &\quad - \int_{\Sigma_{ij}^h} k_\epsilon^{-2} \nabla_{\Sigma_{ij}^h} \varphi_{i(l)}^h \cdot \nabla_{\Sigma_{ij}^h} v_i^h ds \end{aligned} \tag{4.67}$$

where $\boldsymbol{\tau}$ is the unit normal vector to $\partial\Sigma_{ij}^h$. Not only is the boundary term problematic from the computational point of view, but also, it makes the bilinear functional c_{ij}^h non-symmetric. Since the symmetry of c_{ij}^h is necessary to formulate the cross-point algorithm [15], we need to make sure that the boundary term is nullified in a systematic manner. This is easily accomplished if we recall that in the cross-point formulation of DDM, the functional c_{ij}^h acts on functions v_{ij}^h which are non-zero only in the interior of the interface Σ_{ij}^h. Therefore, choosing the test function space for the auxiliary problem (4.60) to be composed of functions which are zero at the cross-points, effectively cancels the boundary contributions. Inadvertently, this also imposes a requirement on the auxiliary functions φ_l to be zero at the boundary $\partial\Sigma_{ij}^h$, since effectively, we apply the cross-point technique to problem (4.52) subject to the problem (4.53) where $\varphi_{(i)l}|_{\Sigma_{ij}^h} = 0$ for $l = 1, \ldots, N_p$. Whether this has any effect on convergence of the resulting method is a challenging problem and requires further investigation.

Structure of the Local Matrices. Before we explain how the cross-point system (4.63) is assembled, we need to establish some notation. To this end, let us introduce the density of descretization h in terms of points per wavelength: $n_\lambda = \lambda/h$. Let us denote by $\mathbf{S}^{\Omega_i^h}$ and $\mathbf{M}^{\Omega_i^h}$ the stiffness and mass matrices for linear elements associated with the domains Ω_i. These matrices have size $N_v^{\Omega_i^h} \times N_v^{\Omega_i^h}$. The constant $N_v^{\Omega_i^h}$ represents the number of degrees of freedom of the finite element space over Ω_i^h. Furthermore, we introduce $\mathbf{S}^{\partial\Omega_i^h}$ and $\mathbf{M}^{\partial\Omega_i^h}$ as the respective matrices stiffness and mass matrices related to the totality of the transmitting surfaces of Ω_i. If these correspond to a generalized stiffness matrix for a surface functions δ, then it is denoted with a subscript $\mathbf{S}_\delta^{\partial\Omega_i^h}$. All these matrices have size $N_v^{\partial\Omega_i^h} \times N_v^{\partial\Omega_i^h}$ where $N_v^{\partial\Omega_i^h}$ is the number of degrees of freedom associated with the transmitting interface. How the mass and stiffness matrices are generated by the finite element method is outlined in Appendix (A.3). Let us denote by $\mathbf{u}_i^{(n+1)} \in \mathbb{C}^{N_v^{\Omega_i^h}}$ to be the local unknown vector and $\boldsymbol{\varphi}_{i(l)}^{(n+1)} \in \mathbb{C}^{N_v^{\Omega_i^h}}$ the surface unknown auxiliary vectors obtained with linear finite element. The discrete test-vectors and right hand side are also bold typed. Then, the discretization of the variational problem for (4.54) - (4.55) yields a coupled linear system:

$$\left(\mathbf{S}^{\Omega_i^h} - k^2\mathbf{M}^{\Omega_i^h} - ikC_0\mathbf{M}^{\partial\Omega_i^h}\right)\mathbf{u}_i^{(n+1)} + ik\sum_{l=1}^{N_p} A_l\mathbf{S}^{\partial\Omega_i^h}\boldsymbol{\varphi}_{i(l)}^{(n+1)} = -\mathbf{M}^{\partial\Omega_i^h}\mathbf{g} \tag{4.68}$$

$$-\mathbf{M}^{\partial\Omega_i^h}\mathbf{u}_i^{(n+1)} - \left(B_l\mathbf{S}_{k_\epsilon^{-2}}^{\partial\Omega_i^h} - \mathbf{M}^{\partial\Omega_i^h}\right)\boldsymbol{\varphi}_{i(l)}^{(n+1)} = \mathbf{0}, \qquad l = 1,\dots,N_p \tag{4.69}$$

The constants C_0, A_l and B_l are the coefficients of the Páde approximation of order N_p to the complex square root function [22]. The square system (4.68) - (4.69) has dimension $N_v^{\Omega_i^h} + N_p\, N_v^{\partial\Omega_i^h}$. Now, the assembly of the system (4.63) involves nothing more than a distribution of the entries of (4.68) and (4.69) into the appropriate matrices $\mathbf{A}_{ii}$, $\mathbf{A}_{ic}$, $\mathbf{A}_{ci}$ and $\mathbf{A}_{cc}$. The process begins by defining vectors $\mathbf{w}_i$ appearing in the matrix Equation (4.63) to have the following structure:

$$\mathbf{w}_i^T = \left(\mathbf{u}_{iI}^{(n+1)}, \boldsymbol{\varphi}_{i(1)}^{(n+1)}, \ldots, \boldsymbol{\varphi}_{i(N_p)}^{(n+1)}\right)^T \tag{4.70}$$

where the subvector $\mathbf{u}_{iI}^{(n+1)}$ contains the unknown nodal values of the solution related to interior and interface degrees of freedom only. Additionally, the vector $\mathbf{w}_c$ of Equation (4.63) contains the global, unknown nodal values of the solution related to the cross-point degrees of freedom. The above Equation (4.70) dictates the structure of the matrices $\mathbf{A}$. For example, matrix $\mathbf{A}_{ii}$ is the restriction of (4.68) - (4.69) to the interior and interface degrees of freedom. It has the following form:

$$\mathbf{A}_{ii} = \left(\begin{array}{c|c|c|c} \mathbf{S}^{\Omega_i^h} - k^2\mathbf{M}^{\Omega_i^h} - ikC_0\mathbf{M}^{\partial\Omega_i^h} & ikA_1\mathbf{S}^{\partial\Omega_i^h} & \ldots & ikA_{N_p}\mathbf{S}_{k_\epsilon^{-2}}^{\partial\Omega_i^h} \\ \hline -\mathbf{M}^{\partial\Omega_i^h} & \mathbf{M}^{\partial\Omega_i^h} - B_1\mathbf{S}_{k_\epsilon^{-2}}^{\partial\Omega_i^h} & \ldots & \mathbf{0} \\ \hline \vdots & \ldots & \ddots & \vdots \\ \hline -\mathbf{M}^{\partial\Omega_i^h} & \mathbf{0} & \ldots & \mathbf{M}^{\partial\Omega_i^h} - B_{N_p}\mathbf{S}_{k_\epsilon^{-2}}^{\partial\Omega_i^h} \end{array}\right) \tag{4.71}$$

Another words, the contributions to the above matrix come from discretization of Equations (4.58) - (4.60) only, while Equation (4.61) contributes nothing. Similarly, the matrix $\mathbf{A}_{ic}$ contains those elements of (4.68) - (4.69) which were obtained by testing interior and interface basis functions against those related to the cross-points. The transpose of that matrix is $\mathbf{A}_{ci}$.

$$\mathbf{A}_{ic} = \left(\begin{array}{c} \mathbf{S}^{\Omega_i^h} - k^2\mathbf{M}^{\Omega_i^h} - ikC_0\mathbf{M}^{\partial\Omega_i^h} \\ \hline \mathbf{0} \\ \hline \vdots \\ \hline \mathbf{0} \end{array}\right) \tag{4.72}$$

The above matrix contains N_p zero vectors corresponding to N_p auxiliary local problems. The only equation contributing to this matrix is Equation (4.61) where again, interior and interface basis functions are tested against those related to the cross-points. The global matrix $\mathbf{A}_{cc}$ is a result of discretization of the same equation. The difference here being that cross-point basis functions are tested against themselves. This matrix is a sum of the entries of local contributions of the system (4.68) restricted to the unknowns related to cross-point nodal values:

$$\mathbf{A}_{cc} = \sum_{i=0}^{N_d-1} \left(\mathbf{S}^{\Omega_i^h} - k^2 \mathbf{M}^{\Omega_i^h} \right) \tag{4.73}$$

4.6 Numerical Results

In this section we apply the domain decomposition algorithm to a 2D model scattering problem truncated with PML. In particular, we will consider the Problem (3.14) of scattering from a sound hard disk centers at the origin. Relative simplicity of the problem domain allows us to write down the solution in the form of a generalized Fourier series given by Equation (3.15). The domain decomposition algorithm described in Section (4.3) will be applied to the corresponding PML problem with the geometry depicted in Figure (4.7) below:

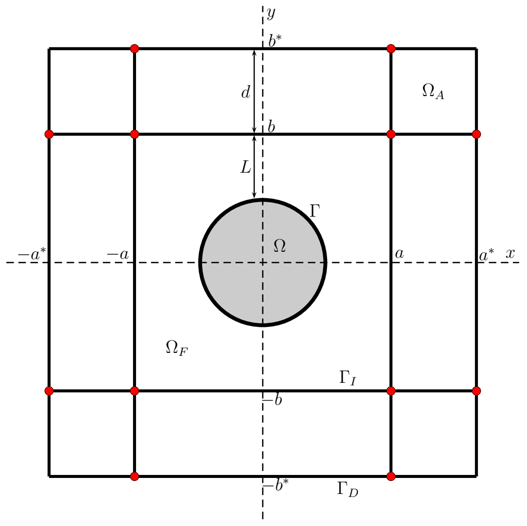


Figure 4.7 Problem domain truncated with PML.

We will study the performance of two transmission operators. First, is the transmission operator used in the Evanescent Mode Damping Algorithm [16, 23]:

$$\Lambda\, u = -ik(1 + i\chi)\, u \tag{4.74}$$

The tables corresponding to the results obtained using the above transmission operator will be denoted by **EMDA**. Second, is the approximation of the Dirichlet-to-Neumann operator for a half-plane (see Section 2.3) obtained by the Padé localization process discussed in Section (2.28). We recall its form below:

$$\Lambda u = -ik\left(C_0 u + \sum_{l=1}^{N_p} A_l \nabla_\Sigma \cdot \left(k_\epsilon^{-2} \nabla_\Sigma\, \varphi_l\right)\right) \tag{4.75}$$

The corresponding tables will be denoted by **PADE**. The auxiliary functions φ_l are coupled to the trace of the solution u through the surface PDEs given by Equations (2.29).

We carry out the simulations using the popular, quadratic absorbing functions with $\sigma^* = 40$ and $p = 2$. The minimal distance from the scatterer to the PML interface Γ_I, denoted by L, is measured in units of wavelength (λ) and since we kept it geometrical units fixed, it changes as we change the wavenumber k. For $k = \pi$, 2π, 3π and 4π it is respectively, $L = 0.5\lambda$, λ, $\frac{3}{2}\pi$, and 2π. On the other hand, we fixed the value of d, which is the width of the PML layer to be always equal to λ, one wavelength. The tables below collect the number of iterations required to attain given residual tolerance as well as the relative error of the approximation.

Table 4.1 EMDA. Number of GMRES Iterations to Attain Residual Tolerance 10^{-6} vs. k for Various Mesh Densities (Number of Points per Wavelength N_λ).

k	$N_\lambda = 12$	$N_\lambda = 16$	$N_\lambda = 20$
π	32	33	40
2π	34	34	39
3π	33	35	37
4π	33	36	37

Table 4.2 EMDA. Relative $l2$-error vs. k (%) for Various Mesh Densities (Number of Points per Wavelength N_λ).

k	$N_\lambda = 12$	$N_\lambda = 16$	$N_\lambda = 20$
π	10	9	7.5
2π	9	7.5	6
3π	9	6.5	5
4π	9	6	4.5

Table 4.3 PADE. Number of GMRES Iterations to Attain Residual Tolerance 10^{-6} vs. k for Various Mesh Densities (Number of Points per Wavelength N_λ).

k	$N_\lambda = 12$	$N_\lambda = 16$	$N_\lambda = 20$
π	26	27	30
2π	23	23	24
3π	20	23	23
4π	19	22	24

Table 4.4 PADE. Relative $l2$-error vs. k (%) for Various Mesh Densities (Number of Points per Wavelength N_λ).

k	$N_\lambda = 12$	$N_\lambda = 16$	$N_\lambda = 20$
π	4	2	1
2π	4.5	2.5	1.5
3π	5	3	2
4π	6	4.5	2

We see that the required number of GMRES iteration so solve the system (4.50) - (4.51) is less when the Padé type transmission condition (4.75) is used. Just as in the case of the original quasi-optimal algorithm [22] where the observed dependence of the number of iterations is weak with respect to the mesh density and the wave number, a similar behavior can be seen here a well. Moreover, the accuracy of the **PADE** method is very good which should not come as a surprise since (4.75) is a more accurate representation of the DtN operator than (4.74).

We also compare the above results with those obtained from truncating the computational domain with a first order radiation condition along Γ_D. No PML is present here. All local problems are Helmholtz boundary value problem of the form (4.50). Especially in the **PADE** case, the error inside of the computational domain Ω_F is much less when the domain is truncated with PML. Also, the number of iterations is slightly lower and varies less with the mesh density.

Table 4.5 EMDA. Number of GMRES Iterations to Attain Residual Tolerance 10^{-6} vs. k for Various Mesh Densities (Number of Points per Wavelength N_λ).

k	$N_\lambda = 12$	$N_\lambda = 16$	$N_\lambda = 20$
π	29	31	36
2π	31	34	36
3π	32	35	38
4π	34	36	39

Table 4.6 EMDA. Relative $l2$-error vs. k at 12 Points per Wavelength (%) for Various Mesh Densities (Number of Points per Wavelength N_λ).

k	$N_\lambda = 12$	$N_\lambda = 16$	$N_\lambda = 20$
π	11	10	9
2π	11	9	8
3π	11	8	7
4π	11	7	7

Table 4.7 PADE. Number of GMRES Iterations to Attain Residual Tolerance 10^{-6} vs. k for Various Mesh Densities (Number of Points per Wavelength N_λ).

k	$N_\lambda = 12$	$N_\lambda = 16$	$N_\lambda = 20$
π	26	32	34
2π	27	33	34
3π	25	30	31
4π	24	26	29

Table 4.8 PADE. Relative $l2$-error vs. k at 12 Points per Wavelength (%) for Various Mesh Densities (Number of Points per Wavelength N_λ).

k	$N_\lambda = 12$	$N_\lambda = 16$	$N_\lambda = 20$
π	5	4.5	4
2π	6.5	5	4.5
3π	8.5	6	5
4π	9	6	5.5

4.7 Conclusions

In this chapter we have presented a new domain decomposition method for the perfectly matched layer system. An analysis of a simple half-plane problem indicates that the convergence rate of the resulting iterative method not only depends on the choice of the transmission operator, but also on the particular PML parameters used. Future investigation of this algorithm should reveal a procedure for determining the optimal parameters. We should mention however, that this problem is difficult in general since one wants to minimize the iteration count while maintaining good accuracy.

Helmholtz boundary value problem is not coercive. This poses challenges when attempting to give general proofs of convergence and accuracy of domain decomposition methods for scattering problems. The presence of non-constant coefficients in the bilinear form corresponding to the local PML problems can be potentially used to force the form to be coercive.

We have presented the method in two dimensional setting. However, an extension to more practical, three dimensional case is straight forward. Indeed, neither the method, nor the algorithm make any assumptions about the dimensionality of the problem. The computational advantage of this approach, as compared with low order radiation conditions should be even more apparent in the higher dimensional case. With the width of the layer kept at one wavelength, the volume of the absorbing layer compared with the volume of the computational domain is very small. The computational work in the layer is therefore negligible. Moreover, as we have seen in the 2D problem, the method achieves very good accuracy with small number of iterations. We would expect the same behavior to persist in the 3D case, although more extensive numerical tests are needed to validate this claim.

CHAPTER 5

IMPROVED TRANSMISSION CONDITIONS FOR DDM IN SCATTERING PROBLEMS

5.1 Motivation

In Chapter 2 we presented a quasi-optimal domain decomposition method based on Padé localization of the square-root transmission operator developed by Boubendir et al. in [22]. This represents the state of the art non-overlapping DDM for the Helmholtz equation. However, to our best knowledge, there is no proof of well-posedness of the local problems. Also, as we shall explain in the later section, we observe that the size of the local problems grows with the order of the Padé series. In this chapter, we introduce a new family of transmission conditions which doesn't suffer from these drawbacks. We show that when used in the context of Padé type localization of the square-root operator, the local problems are well-posed and their size remains independent of the order of the approximating series. The numerical study of the new algorithm shows that the method achieves the same accuracy at the fraction of the computational time.

5.2 Improved Transmission Conditions

To derive the new algorithm we rewrite the continuity conditions (2.5) - (2.6) in the following equivalent form:

$$\partial_{\nu_i} u_i + \alpha u_i + T_i\, u_i = -\partial_{\nu_j} u_j + \alpha u_j + T_i\, u_j \tag{5.1}$$

$$\partial_{\nu_j} u_j + \alpha u_j + T_j\, u_j = -\partial_{\nu_i} u_i + \alpha u_i + T_j\, u_i \tag{5.2}$$

where $\alpha \in \mathbf{C}$ is a constant and T_i, T_j are operators to be defined later. The first step in defining the new transmission condition involves moving the terms containing T_i

and T_j to the right hand side:

$$\partial_{\nu_i} u_i + \alpha u_i = -\partial_{\nu_j} u_j + \alpha u_j + T_i \left(u_j - u_i\right) \tag{5.3}$$

$$\partial_{\nu_i} u_j + \alpha u_j = -\partial_{\nu_i} u_i + \alpha u_i + T_j \left(u_i - u_j\right) \tag{5.4}$$

We now define an iteration procedure in which the following problems are solved at each step:

$$\begin{aligned} \Delta u_i^{(n+1)} + k^2 u_i^{(n+1)} &= 0 \quad \text{in} \quad \Omega_i \\ \partial_{\nu_i} u_i^{(n+1)} &= \partial_{\nu_i} u^i \quad \text{on} \quad \Gamma_i \\ \partial_{\nu_i} u_i^{(n+1)} - iku_i^{(n+1)} &= 0 \quad \text{on} \quad \Sigma_i \end{aligned} \tag{5.5}$$

$$\partial_{\nu_i} u_i^{(n+1)} + \alpha u_i^{(n+1)} = g_{ij}^{(n)} \quad \text{for} \quad \Sigma_{ij} \tag{5.6}$$

where:

$$g_{ij}^{(n)} = -\partial_{\nu_j} u_j^{(n)} + \alpha u_j^{(n)} + T_i \left(u_j^{(n)} - u_i^{(n)}\right) \tag{5.7}$$

is the interface datum corresponding to an interface Σ_{ij}. Introduction of an iterative scheme allowed us to relax the continuity conditions (5.1)-(5.2) and close the local problems with the transmission condition (5.6). The update equation is determined the same way as in Chapter 2. A simple algebraic manipulation yields:

$$g_{ij}^{(n+1)} = -g_{ji}^{(n)} + 2\alpha\, u_i^{(n+1)} + T_i \left(u_j^{(n)} - u_i^{(n)}\right) \tag{5.8}$$

As we can see, α could be chosen so that the local problems are always well-posed. Furthermore, the operator T_i could be non-local without affecting the cost and conditioning of the local problems. In what follows, we are going to describe the new algorithm in a particular case where the operators T are the Padé approximations to the the square-root operator used previously.

To that end, let us recall the form of the transmission condition used in the quasi-optimal DDM. For an interface Σ_{ij} it is given by the equation below:

$$\partial_{\nu_i} u_i - ikC_0 u_i - ik \sum_{l=1}^{N_p} A_l \, S_{ij} \, u_i = -\partial_{\nu_j} u_j - ikC_0 u_j - ik \sum_{l=1}^{N_p} A_l \, S_{ij} \, u_j \tag{5.9}$$

where the operators S_{ij} are defined by:

$$S_{ij} = \nabla_{\Sigma_{ij}} \cdot \left(k_\epsilon^{-2} \nabla_{\Sigma_{ij}}\right) \left(1 + B_l \nabla_{\Sigma_{ij}} \cdot \left(k_\epsilon^{-2} \nabla_{\Sigma_{ij}}\right)\right)^{-1}. \tag{5.10}$$

Comparing Equations (5.9) and (5.1) we immediately see that the two are equivalent provided that we make following definitions for α and T_i:

$$\alpha = -ikC_0 \tag{5.11}$$

$$T_i = -ik \sum_{l=1}^{N_p} A_l \, S_{ij} \tag{5.12}$$

The well-posedness of local problems with the above choices for α and T_i is relatively easy to establish. C_0 is a complex constant since it is the coefficient of the rational approximation of the square root function (2.26). The homogeneous local problems therefore, have the same structure as those studied by Boubendir in [15] in the context of EMDA technique. Their well-posedness have been established in that same reference.

The presence of the inverse of a differential operator in S_{ij} is inconvenient. The make the derivation of a variational formulation of the resulting local problems easier, we introduce auxiliary functions $\varphi_{i(l)}$. For each interface Σ_{ij}, they are the solutions of the surface PDEs:

$$\left(1 + B_l \nabla_{\Sigma_{ij}} \cdot \left(k_\epsilon^{-2} \nabla_{\Sigma_{ij}}\right)\right) \varphi_{i(l)} = u_i \tag{5.13}$$

With their help, we can rewrite Equation (5.12) defining the operator T_i as:

$$T_i\, u_i = -ik \sum_{l=1}^{N_p} A_l\, \nabla_{\Sigma_{ij}} \cdot \left(k_\epsilon^{-2} \nabla_{\Sigma_{ij}}\, \varphi_{i(l)} \right) \tag{5.14}$$

Since the problems (5.13) are linear, the right hand side g_{ij} of the transmission condition (5.6) can be conveniently written as:

$$g_{ij} = -\partial_{\nu_j} u_j - ikC_0 u_j - ik \sum_{l=1}^{N_p} A_l \nabla_{\Sigma_{ij}} \cdot \left(k_\epsilon^{-2} \nabla_{\Sigma_{ij}} \left[\varphi_{j(l)} - \varphi_{i(l)} \right] \right) \tag{5.15}$$

It should be noted that in this formulation of local problems, the auxiliary functions $\varphi_{i(l)}$ are no longer coupled to the local solution u_i. This is in contrast to the case of the quasi-optimal algorithm described in Chapter 2. This is easy to see once we realize that at step $n+1$, the function g_{ij} is evaluated at index n. Consequently, the size of the resulting local matrices is exactly the same as in the case of the original algorithm due to Déspres [33]. The update and exchange of interface data is slightly more involved however. For each subdomain, matrices related to finite element discretization of Equation (5.13) need to inverted. Since the number of interface unknowns is relatively small compared to the total number of unknowns in the system, these matrices can be pre-factorized and stored in memory during the initialization phase. In this case the exchange procedure involves only a simple back substitution process which is very efficient.

5.3 Discussion of the Structure of Local Matrices

Let us consider the 2D model scattering problem (2.1) of Section (2.1). The quasi-optimal domain decomposition method for that problem was presented in the subsequent Section (2.2). In that formulation of the method, the local problems are given by Equations (2.30) - (2.34). The nodal finite element method applied to these problems yields systems of equations of the following form:

$$\left(\mathbf{S}^{\Omega_i^h} - k^2\mathbf{M}^{\Omega_i^h} - ikC_0\mathbf{M}^{\partial\Omega_i^h}\right)\mathbf{u}_i^{(n+1)} + ik\sum_{l=1}^{N_p} A_l \mathbf{S}^{\partial\Omega_i^h}\boldsymbol{\varphi}_{i(l)}^{(n+1)} = -\mathbf{M}^{\partial\Omega_i^h}\mathbf{g} \qquad (5.16)$$

$$-\mathbf{M}^{\partial\Omega_i^h}\mathbf{u}_i^{(n+1)} - \left(B_l\mathbf{S}^{\partial\Omega_i^h}_{k_\epsilon^{-2}} - \mathbf{M}^{\partial\Omega_i^h}\right)\boldsymbol{\varphi}_{i(l)}^{(n+1)} = \mathbf{0}, \qquad l = 1, \ldots, N_p \qquad (5.17)$$

where by $\mathbf{S}^{\Omega_i^h}$ and $\mathbf{M}^{\Omega_i^h}$ respectively, we denote the stiffness and mass matrices for linear elements associated with the domains Ω_i. These matrices have size $N_v^{\Omega_i^h} \times N_v^{\Omega_i^h}$. The constant $N_v^{\Omega_i^h}$ represents the number of degrees of freedom of the finite element space over Ω_i^h. Furthermore, we introduce $\mathbf{S}^{\partial\Omega_i^h}$ and $\mathbf{M}^{\partial\Omega_i^h}$ as the respective matrices stiffness and mass matrices related to the totality of the transmitting surfaces of Ω_i. If these correspond to a generalized stiffness matrix for a surface functions δ, then it is denoted with a subscript $\mathbf{S}_\delta^{\partial\Omega_i^h}$. All these matrices have size $N_v^{\partial\Omega_i^h} \times N_v^{\partial\Omega_i^h}$ where $N_v^{\partial\Omega_i^h}$ is the number of degrees of freedom associated with the transmitting interface. How the mass and stiffness matrices are generated by the finite element method is outlined in Appendix (A.3). Let us denote by $\mathbf{u}_i^{(n+1)} \in \mathbb{C}^{N_v^{\Omega_i^h}}$ to be the local unknown vector and $\boldsymbol{\varphi}_{i(l)}^{(n+1)} \in \mathbb{C}^{N_v^{\Omega_i^h}}$ the surface unknown auxiliary vectors obtained with linear finite element. The discrete test-vectors and right hand side are also bold typed.

The strength of the Padé localization technique lies in the fact that it approximates the Dirichlet-to-Neumann operator by local, differential operators uniformly over all wavelengths. Consequently, an accurate representation of the operator can be obtained by finite element discretization which yields sparse instead of dense matrices as in the case of boundary element method. The size of the local finite element matrices however, is proportional to the order of the Padé approximation used. This can potentially lead to prohibitively large linear problems that need to be solved at each iteration.

This claim can be easily visualized by observing the structure of the linear system given by Equations (5.16) - (5.17). This square system has dimension

$N_v^{\Omega_i^h} + N_p N_v^{\partial\Omega_i^h}$. It is clear that the size of the system grows as the number of unknowns increases, or as the finite element mesh is refined, especially in three-dimensional geometries. More importantly, higher values of the order of the Páde series N_p also result in larger local systems needed to be assembled and factorized. Mesh partitioning could be used to decrease the number of degrees of freedom by making the local domain Ω_i smaller. Such strategy however, does not guarantee the decrease in the computational time of the resulting parallel iterative solver due to the associated communication overhead. In fact, the well known Amdahl's law states that the speed-up of a parallel application is linear only for small number of the processing units utilized. It is not to say that the use of large number of subdomains does not result in the decrease of the computational time. Rather, to quantify the potential speedup is very difficult in a practical setting. Moreover, as we show shortly, the use of the modified condition results in smaller local systems and an iterative method with satisfactory convergence rates. In fact, our numerical study shows that the use of the transmission condition proposed here results in a method whose convergence properties are comparable to those of the quasi-optimal method presented in [22], and which achieves the same accuracy at a much lower computational cost.

5.4 Numerical Results

In this section we study the performance of the new transmission condition. We compare it with the quasi-optimal domain decomposition method. We are interested not only in the iteration count and accuracy, but also in the computational time required to obtain the solution. In 3D setting, the quasi-optimal method can be costly, especially when the order N_p of the approximating series is high. We will show that the transmission conditions proposed in this chapter achieves small iteration count, good accuracy and is computationally much more tractable.

2D Problem We begin with a simple 2D model problem of scattering of time-harmonic plane acoustic waves from a sound hard unit disk centered at the origin. The problem geometry is shown below (Figure 5.1). The artificial, circular boundary Σ with three unit radius is centered at the origin as well.

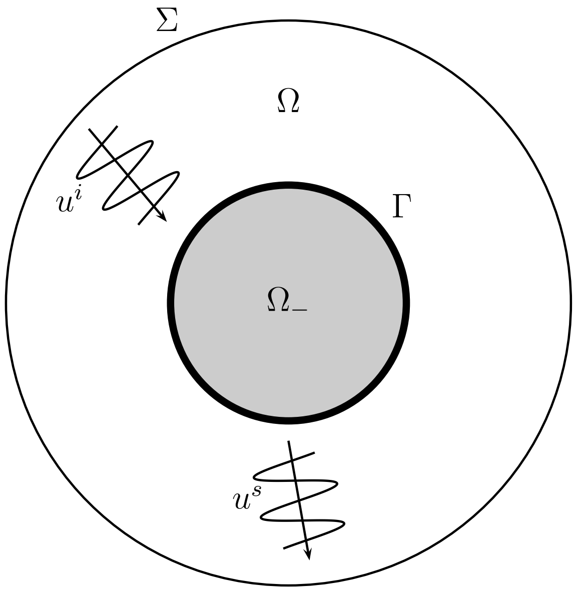


Figure 5.1 Two-dimensional geometry of the model problem.

We are interested in obtaining the amplitude of the scattered field u governed by:

$$\begin{cases} \Delta u + k^2 u = 0 \quad \text{in} \quad \Omega \\ \partial_r u = -\partial_r \, e^{-ikx} \quad \text{on} \quad \Gamma \\ \partial_r u - iku = 0 \quad \text{on} \quad \Sigma \end{cases} \tag{5.18}$$

We apply DDM to the partition of Ω into five non-overlapping, washer-like subdomains $\Omega_i : i = 1, \ldots, 5$. The four artificial circular interfaces have radii R_j equal to 1.4, 1.8, 2.2, 2.6 units respectively. In order to calculate the Páde coefficients we fix the order of approximation $N_p = 8$ and the parameter $\theta = \pi/4$. Also, we let $k_\epsilon = k + i0.4k^{1/3}R_j^{-2/3}$ where R_j is the radius of the circular interface. These quantities have been found numerically to produce good results [22]. The accuracy of the method is assessed by comparing with the true solution which can be written in the following form:

$$u(r,\theta) = \sum_{n=-\infty}^{\infty} \left[\alpha_n H_n^{(1)}(kr) + \beta_n H_n^{(2)}(kr)\right] e^{in\theta} \tag{5.19}$$

where $H_n^{(1)}(z)$ and $H_n^{(2)}(z)$ are the Hankel functions of first and second kind respectively. The coefficients α_n and β_n are the solutions of the system of equations given below:

$$\begin{bmatrix} d_z H_n^{(1)}(ka) & d_z H_n^{(2)}(ka) \\ d_z H_n^{(1)}(kb) - iH_n^{(1)}(kb) & d_z H_n^{(1)}(kb) - iH_n^{(1)}(kb) \end{bmatrix} \begin{bmatrix} \alpha_n \\ \beta_n \end{bmatrix} = \begin{bmatrix} i^n d_z J_n(ka) \\ 0 \end{bmatrix} \tag{5.20}$$

Function $J_n(z)$ is the Bessel function of the first kind. We compare the proposed algorithm (denoted by **NEW PADE**) with the quasi-optimal Páde method [22] (denoted by **PADE**) with the same parameters.

Table 5.1 Iteration Count vs. k at 12 Points per Wavelength.

Residual Tolerance 10^{-6}.

k	**NEW PADE**	**PADE**
π	38	30
2π	33	26
3π	32	27
4π	34	28

Table 5.2 Relative $l2$-error vs. k at 12 Points per Wavelength (%).

k	**NEW PADE**	**PADE**
π	5.6	3.6
2π	7.4	6.4
3π	11.5	10.0
4π	13.6	13.3

Table 5.3 Computation Time vs. k at 12 Points per Wavelength (sec).

k	**NEW PADE**	**PADE**
π	2.4	2.2
2π	8.0	7.5
3π	22.0	21.2
4π	51.5	50.2

Table 5.4 Iteration Count vs. k at 20 Points per Wavelength.

Residual Tolerance 10^{-6}.

k	**NEW PADE**	**PADE**
π	39	30
2π	35	26
3π	32	26
4π	35	28

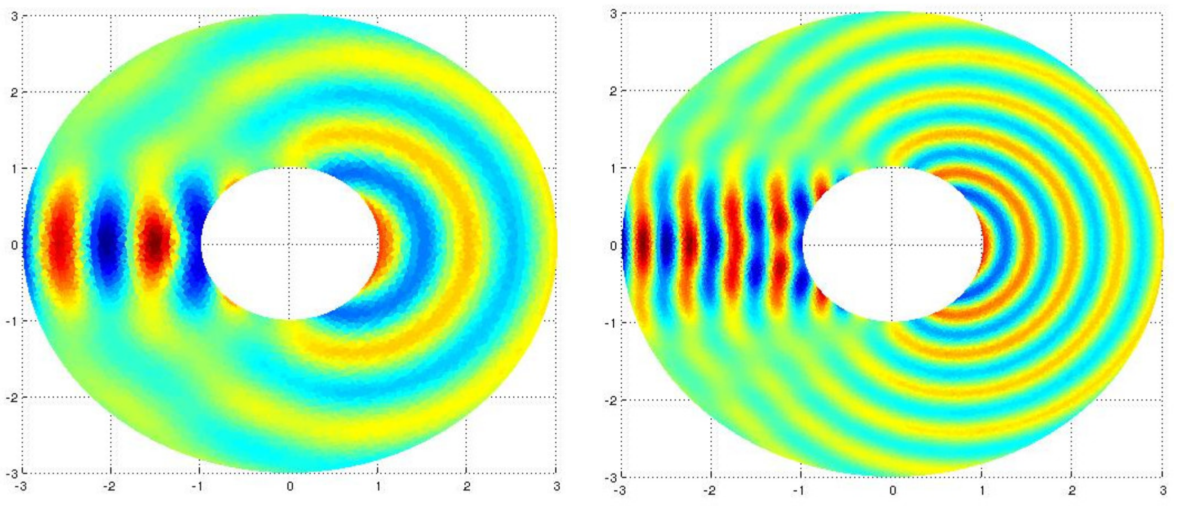
Table 5.5 Relative $l2$-error vs. k at 20 Points per Wavelength (%).

k	**NEW PADE**	**PADE**
π	2.4	1.5
2π	2.8	2.4
3π	4.6	4.0
4π	5.1	4.9

Table 5.6 Computation Time vs. k at 20 Points per Wavelength (sec).

k	**NEW PADE**	**PADE**
π	5.8	5.2
2π	30.8	28.7
3π	103.0	103.0
4π	295.0	290.0

The 2D simulations do not reveal the practicality of the new method since the total number of unknowns and degrees of freedom involved in the exchange are relatively low. In this case, the performance of the new transmission condition is comparable with the quasi-optimal method. Both methods do perform better than EMDA, which however is easier to implement. The figure below shows the amplitude u of the scattered field at $k = 2\pi$ and 4π respectively.



(a) $k = 2\pi$ **(b)** $k = 4\pi$

Figure 5.2 Plot of the numerical solution at different wavelengths.

3D Problem Let us now consider a 3D model problem of acoustic scattering from a sound hard sphere centered at the origin. We can think of Figure (5.1) as showing a cross section of the computational domain with the cut plane passing through the origin. Domain decomposition method will be applied to the 3D counterpart of Problem (5.18). The true solution to the problem can be obtained by the method of separation of variables. With the vector of incidence pointing in the negative z-direction, the solution can be written as a series expansion in spherical harmonics [48]:

$$u(r,\theta,\phi) = \sum_{n=0}^{\infty} \left[\alpha_n H_n^{(1)}(kr) + \beta_n H_n^{(2)}(kr)\right] P_n^0(\cos\theta) \tag{5.21}$$

where the coefficients α_n and β_n are given by the solution of the following system of equations:

$$\begin{bmatrix} d_z H_n^{(1)}(ka) & d_z H_n^{(2)}(ka) \\ d_z H_n^{(1)}(kb) - iH_n^{(1)}(kb) & d_z H_n^{(1)}(kb) - iH_n^{(1)}(kb) \end{bmatrix} \begin{bmatrix} \alpha_n \\ \beta_n \end{bmatrix} = \begin{bmatrix} i^n(2n+1) d_z J_n(ka) \\ 0 \end{bmatrix} \tag{5.22}$$

$H^{(1)}(z)$ and $H^{(2)}(z)$ are the first and second kind spherical Hankel functions, $P_n^0(\cos\theta)$ is the associated Legendre function, and $J_n(z)$ is the Bessel function of the first kind.

Decomposition Without Cross-Points We first consider partitions of the domain containing no cross-points. We will decompose the volumetric domain Ω into 3, 5 and 7 spherical shells shown in the Figures (5.3) - (5.5) below. At each wavelength $k = 2\pi$, 4π and 8π, we report the the number of iterations required to attain prescribed residual tolerance, relative error as compared with the true solution (5.21) and various timings at 10 and 20 points per wavelength for the domain decomposition methods proposed in this chapter, denoted by **NEW PADE**, as well as the quasi-optimal method discussed earlier (see Chapter 2), denoted by **PADE**.

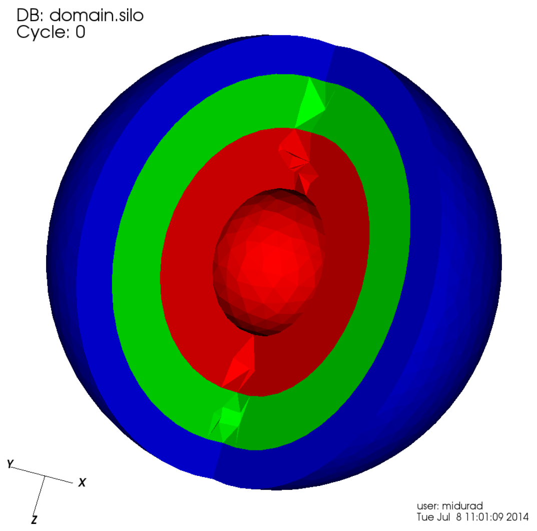


Figure 5.3 3D spherical geometry partitioned into **3 subdomains**.

Table 5.7 Number of Iterations Required vs. k at $N_\lambda = 10$, 3 Subdomains.

k	NEW PADE	PADE
2π	29	23
4π	27	22
8π	26	20

Table 5.8 Relative Error vs. k (%) at $N_\lambda = 10$, 3 Subdomains.

k	**NEW PADE**	**PADE**
2π	15	15
4π	17	17
8π	30	30

Table 5.9 Total Computation Time vs. k (min) at $N_\lambda = 10$, 3 Subdomains.

k	**NEW PADE**	**PADE**
2π	0.6	2.5
4π	1.8	39
8π	18	668

Table 5.10 Number of Iterations Required vs. k at $N_\lambda = 20$, 3 Subdomains.

k	**NEW PADE**	**PADE**
2π	31	22
4π	33	24
8π	31	27

Table 5.11 Relative Error vs. k (%) at $N_\lambda = 20$, 3 Subdomains.

k	**NEW PADE**	**PADE**
2π	4	4
4π	4	4
8π	8	8

Table 5.12 Total Computation Time vs. k (min) at $N_\lambda = 20$, 3 Subdomains.

k	**NEW PADE**	**PADE**
2π	2	39
4π	17	660
8π	1150	5345

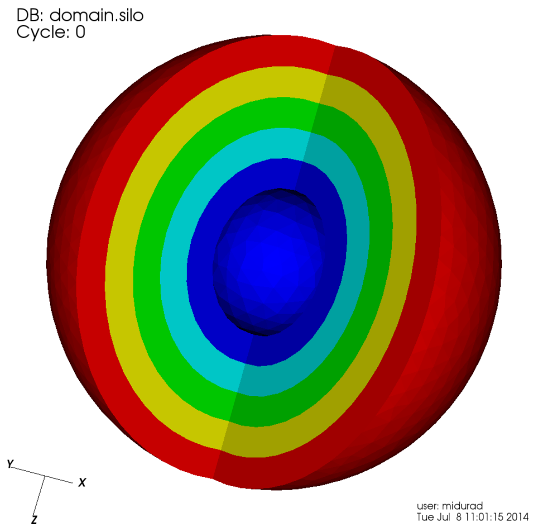


Figure 5.4 3D spherical geometry partitioned into 5 subdomains.

Table 5.13 Number of Iterations Required vs. k at $N_\lambda = 10$, 5 Subdomains.

k	**NEW PADE**	**PADE**
2π	90	90
4π	102	102
8π	63	48

Table 5.14 Relative Error vs. k (%) at $N_\lambda = 10$, 5 Subdomains.

k	**NEW PADE**	**PADE**
2π	10	10
4π	15	15
8π	29	29

Table 5.15 Total Computation Time vs. k (min) at $N_\lambda = 10$, 5 Subdomains.

k	**NEW PADE**	**PADE**
2π	1.2	4.5
4π	4.6	69
8π	15	962

Table 5.16 Number of Iterations Required vs. k at $N_\lambda = 20$, 5 Subdomains.

k	**NEW PADE**	**PADE**
2π	55	37
4π	51	40
8π	40	35

Table 5.17 Relative Error vs. k (%) at $N_\lambda = 20$, 5 Subdomains.

k	**NEW PADE**	**PADE**
2π	4	4
4π	5	5
8π	8	8

Table 5.18 Total Computation Time vs. k (min) at $N_\lambda = 20$, 5 Subdomains.

k	**NEW PADE**	**PADE**
2π	2.5	48
4π	16	805
8π	450	6537

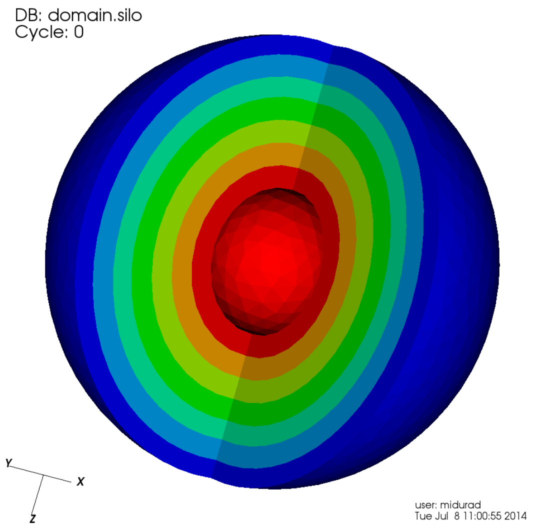


Figure 5.5 3D spherical geometry partitioned into 7 subdomains.

Table 5.19 Number of Iterations Required vs. k at $N_\lambda = 10$, 7 Subdomains.

k	**NEW PADE**	**PADE**
2π	75	66
4π	46	37
8π	38	29

Table 5.20 Relative Error vs. k (%) at $N_\lambda = 10$, 7 Subdomains.

k	**NEW PADE**	**PADE**
2π	14	14
4π	19	19
8π	32	32

Table 5.21 Total Computation Time vs. k (min) at $N_\lambda = 10$, 7 Subdomains.

k	**NEW PADE**	**PADE**
2π	0.9	3.4
4π	2.2	45
8π	15	688

Table 5.22 Number of Iterations Required vs. k at $N_\lambda = 20$, 7 Subdomains.

k	**NEW PADE**	**PADE**
2π	110	88
4π	68	51
8π	62	48

Table 5.23 Relative Error vs. k (%) at $N_\lambda = 20$, Subdomains.

k	**NEW PADE**	**PADE**
2π	4	4
4π	4	4
8π	8	8

Table 5.24 Total Computation Time vs. k at $N_\lambda = 20$, 7 Subdomains.

k	**NEW PADE**	**PADE**
2π	5	70
4π	16	900
8π	185	7567

The accuracy of both methods is the same. Of course, the error is smaller in the case of 20 points per wavelength. The quasi-optimal method (**PADE**) requires less iterations to converge to a residual error tolerance of 10^{-6}. However, a single iteration of this method requires more work than that of the method proposed in this chapter (**NEW PADE**). The initialization that includes the assembly and factorization of local matrices is also more expensive in the latter case as explained in Section (5.3). Tables that present the total running time of the algorithms show that the new transmission condition results in a much more computationally effective algorithm even though the iteration count is slightly higher. Next, we present few results of the method applied to decompositions with cross-points. The same technique of adapting the quasi-optimal method to the cross-point technique presented in Chapter 4 will be used next.

Decomposition With Cross-Points We have also considered a partition of the computational domain that contains cross points. The Figure (5.6) below shows a one half of the domain partitioned into four subdomains. The other half is symmetric about the cut plane. The green points show location of the cross-points.

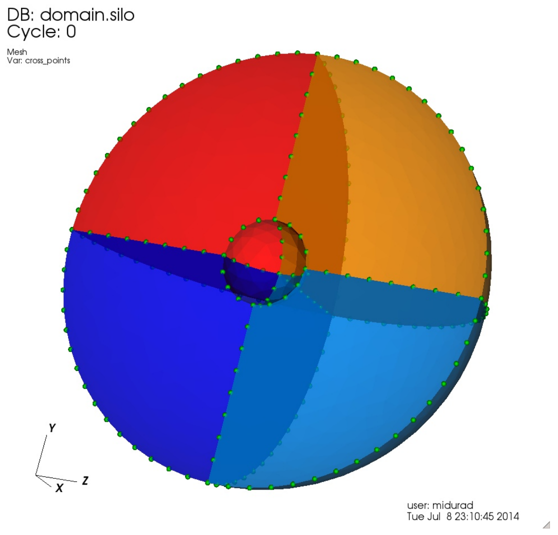


Figure 5.6 3D spherical geometry partitioned into 8 subdomains.

Table 5.25 Iteration Count vs. k at $N_\lambda = 10$.

k	**NEW PADE**	**PADE**
π	21	20
2π	37	31
4π	50	50
8π	56	54

Table 5.26 Relative $l2$-error vs. k at $N_\lambda = 10$ (%).

k	**NEW PADE**	**PADE**
π	36	36
2π	14	14
4π	16	16
8π	28	28

Table 5.27 Computation Time vs. k at $N_\lambda = 10$ (min).

k	**NEW PADE**	**PADE**
π	0.35	0.35
2π	0.42	0.47
4π	1.14	1.63
8π	8.53	22.83

Table 5.28 Iteration Count vs. k at $N_\lambda = 20$.

k	**NEW PADE**	**PADE**
π	31	25
2π	47	40
4π	62	51
8π	57	66

Table 5.29 Relative $l2$-error vs. k at $N_\lambda = 20$ (%).

k	**NEW PADE**	**PADE**
π	10	10
2π	5	5
4π	5	5
8π	8	8

Table 5.30 Computation Time vs. k at $N_\lambda = 20$ (min).

k	**NEW PADE**	**PADE**
π	0.42	0.45
2π	1	1.47
4π	9	22
8π	478	927

Again, the accuracy of both methods is the same. The computational domain used in this case is identical to that of the previous section. The only difference is the way it has been decomposed. Of course, since there are cross-points, the resulting iterative method is that given by the cross-point technique. Here however, we do observe a stronger dependence of the iteration count on the wavenumber k. Nevertheless, the computational time of the new algorithm remains relatively small while preserving good accuracy of the solution.

5.5 Conclusions

In this chapter we have presented a new family of transmission conditions for domain decomposition methods for the scattering problems. The modification of the usual transmission conditions amounts to an introduction of a *correcting* term. The strength of this approach lies in the realization that this term can be acted upon by an operator which in principle, could be chosen to improve convergence as well as efficiency of the resulting method. In this chapter, the choice of this operator have been motivated by the structure of the transmission condition obtained from the localization of the square-root operator. We were able to show numerically, that the new method resulted in an efficient algorithm with very good accuracy. As mentioned in the opening section of this chapter, the form of the transmission conditions proposed here makes it possible to give proofs of well-posedness of the local problems. This is quite easy to see if we realize that local homogeneous problems have the same form as those analyzed initially be B. Després. Further investigation is needed to determine the properties of the operator T and the accompanying constant α that would improve convergence as well as their response to varying problem parameters such as the wavenumber, local problem geometry or number of point per wavelength.

CHAPTER 6

PARALLEL IMPLEMENTATION

In this chapter, we give an overview of data structures and algorithms necessary for an efficient, parallel implementation of the domain decomposition methods described thus far. We assume that our computational model is that of a single program and multiple data (SPMD). Such computing paradigm is usually realized via message passing on a distributed memory computer architectures. Clusters of computers, connected to a dedicated, high speed network conforming to the Message Passing Interface (MPI) specifications [38], are a prime example of such architectures. The MPI programming model is now a standard in high performance computing community. By abstracting away details related to low level network programming, it allows the software developer to focus exclusively on the logic of data flow in the cluster, and that greatly reduces the complexity of the resulting code.

The challenge to writing a parallel scientific application of this kind is two-fold. Firstly, the underlying data structures have to be conceived totally differently. Secondly, the algorithm and data structure design is not trivial when the goals is to achieve scalability and extensibility. There exist many excellent finite element libraries, but only few meet the above criteria. The deal.ii [8, 9] and Dune [10, 11] libraries seems most mature in that respect. They are designed for massively parallel computations from the ground up using modern C++ programming techniques. They are also very big systems, promising to solve a wide variety of physical problems using the finite element method. Unfortunately, we were unable to find an efficient way to query their mesh objects for the cross-point nodes. For the purposes of our algorithms, this operation is essential and we were therefore compelled to build our own parallel, finite element system. Many design decisions we have made were influenced by both of the libraries just mentioned.

In the following sections, we shall look at the problems of scalability and extensibility and how they relate to the implementation of our algorithms in more detail. We begin with a list of necessary requirements that ought to be satisfied by our system and an overview of the resulting architecture.

6.1 System Requirements and Architecture

Developing extensible finite element software is not trivial [34,35,77]. The goal of the resulting framework is to support implementation and testing of a variety of domain decomposition algorithms. There are many decisions that need to be made before the development starts. Are we going to solve 2D, 3D problems or both? What type of a mesh? Triangular, tetrahedral, quadrilateral? What finite element spaces are we going to use? What quadrature rules will be most applicable? How are we going to solve the resulting linear system? It is very difficult to give an answer to these questions in advance. It is therefore imperative to approach the problem from a modular perspective.

Modular approach to software design has many advantages. Every component is seen as a black box by every other component in the system. The only way components can communicate is through a well defined application programming interface (API). The benefit of such a design is the ability to extend and modify the system relatively easily without disrupting the whole.

The major components found in our framework are shown in the Figure (6.1) below:

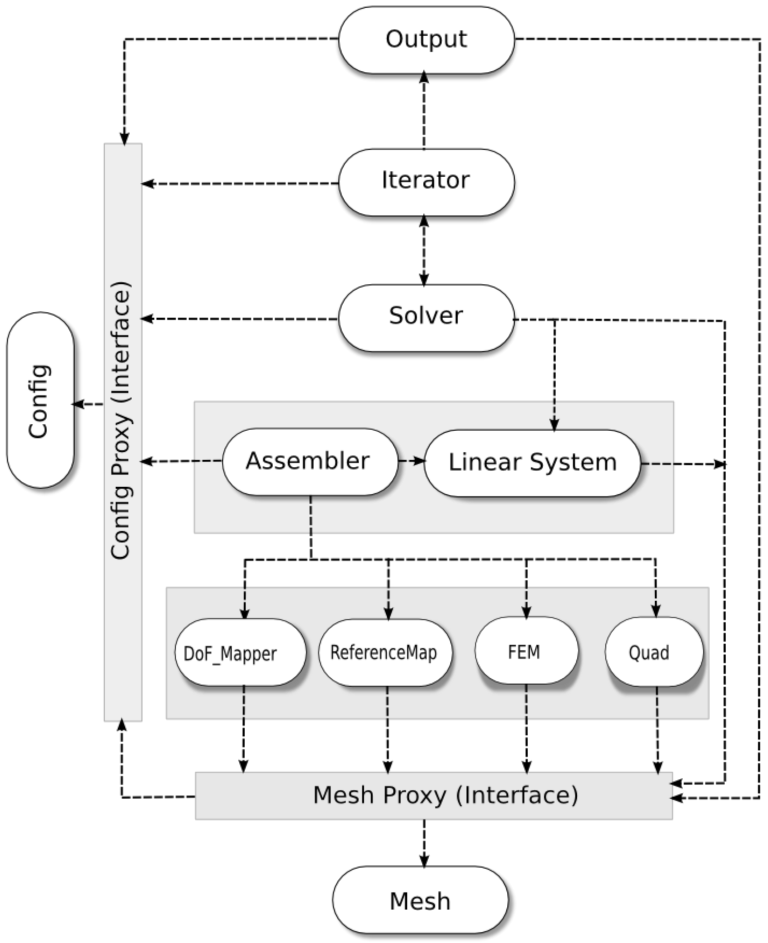


Figure 6.1 System components. Arrows point in the direction of dependency.

Component Description:

- *Mesh* - Representation of the distributed mesh. It conforms to the mesh API implemented by the (*Mesh Proxy*).
- *Mesh Proxy* - Encapsulates the mesh API. Hides details of the mesh representation from the user of the system.

- *DoF_Mapper* - Provides various sets of indices necessary in finite element and domain decomposition computations such as mappings from global vertex IDs to local (partition) vertex IDs, etc.
- *ReferenceMap* - Defines reference cells/elements and the corresponding mappings.
- *FEM* - Holds the definitions of shape functions and their gradients on reference elements/cells.
- *Quad* - Defines quadrature weights and nodes for a variety of reference elements.
- *Assembler* - Encapsulates a user defined assembly routine.
- *Linear System* - Holds a linear system associated with a given problem. Also user defined.
- *Solver* - This component abstracts away the paralell nature of the *Linear System* by providing an intuitive interface to the iteration operator having the semantics of matrix-vector multiplication.
- *Iterator* - An abstract object which implements an iterative method used to solve the iteration equations associated with the domain decomposition algorithm.
- *Output* - A component responsible for gathering of the solution and outputting it into a user specified data format.
- *Config* - Holds parameters relevant to the problem being solved such as the wave number, number of mesh partitions, etc.
- *Config Proxy* - Encapsulates the config API.

The major requirements we want our framework to satisfy are as follows:

1. For efficiency, the system should be built on top of a distributed mesh, automatically partitioned with automatic cross-point detection.

2. The API should allow to improve existing, or add new system features easily.

3. Variation formulations are dimension independent. We want to express our algorithms in the same way.

4. The API should hide the details involved in the calculation of the action of the iteration operator and provide an interface with semantics of matrix-vector multiplication to that operation.

The first item on the list has to do with scalability. One cannot hope to perform massively parallel simulation when the global mesh data is replicated through the cluster. The only way to handle very large problems is to partition the mesh and distribute the pieces to the corresponding cluster nodes. This way, a single node holds only that portion of the mesh which is associated with the local problem assigned to it. This reduces the memory footprint of the mesh object but also requires a more advanced data structure which we describe in the next section.

The second item pertains a good API which is relatively easy to define once the system has been decomposed into components with well defined responsibilities. The mesh API reflects the topological structure of the mesh, which can be thought of as a collection of mesh entities of lower dimensions [13]. A distributed mesh is a collection of partitions. Each partition is a collection of cells and interfaces. Here a geometric representation of a cell depends on the type of the mesh and the dimension of the space it is embedded in. Cells could be triangles, quadrilaterals, etc. in 2D or tetrahedral, prisms, etc. in 3D. A cell in turn, is a collection of faces. Those could correspond to triangle edges, or tetrahedral faces, depending on the kind of the mesh being represented. An interface is a collection of faces common to two neighboring partitions. A face is a collection of vertices which represent points in space. Geometrically, a face is a convex hull of it vertices. The assembly of discrete, finite element equations as well as the update and exchange steps in the domain decomposition algorithms require traversal of many of these collections. Programmatically, traversals are most conveniently expressed in terms of iterators [66]. The concept of an iterator is language agnostic, but since we have implemented our code in C++, the code snippets below show iterator semantics found in the C++'s Standard Template Library (STL) [66]. The resulting high level interface encapsulated by the *Mesh Proxy* component exposes the following abstract data types:

```
template<int wdim>
class Vertex {
        int id;

        Point<wdim> coordinates();

        bool isFree();
        bool isConstrained();
        bool isCrossPoint();
        ...
};
```

```
template<int wdim>
class Face {
        HalfFace<wdim> id;

        Partition<wdim>* parentPartition;

        VertexIterator<wdim> beginVertices();
        VertexIterator<wdim> endVertices();
        ...
};
```

```
template<int wdim>
class Cell {
        int id;
        enum CELL_TYPE;

        Partition<wdim>* parentPartition;

        FaceIterator<wdim> beginFaces();
        FaceIterator<wdim> endFaces();
        VertexIterator<wdim> beginVertices();
        VertexIterator<wdim> endVertices();
        ...
};
```

```
template<int wdim>
class Interface {
        Partition<wdim>* parentPartition;
        int neighborRank;

        FaceIterator<wdim> beginFaces();
        FaceIterator<wdim> endFaces();
        VertexIterator<wdim> beginVertices();
        VertexIterator<wdim> endVertices();
        ...
};
```

```
template<int wdim>
class Partition {
        PartitionRaw* partitionRaw;

        int myRank;
        int meshRank;

        InterfaceIterator<wdim> beginInterfaces();
        InterfaceIterator<wdim> endInterfaces();
        CellIterator<wdim> beginCells();
        CellIterator<wdim> endCells();
        FaceIterator<wdim> beginFaces();
        FaceIterator<wdim> endFaces();
        VertexIterator<wdim> beginVertices();
        VertexIterator<wdim> endVertices();
        ...
};
```

```
template<int wdim>
class Mesh {
        vector<int> partitionRanks;
        ParallelNeighborhood parallelNbd;
        ...
};
```

The integer template parameter *wdim* is used to specify the dimension of the problem domain. That is, *wdim = 2* for 2D problems, and *wdim = 3* for 3D problems. Such use of C++ template system allows us to implement finite element formulations in a dimension independent way, which takes care of the third requirement. The integer *rank* variables are used for communication within the MPI context. The *rank* is also an id of the corresponding partition. Each iterator extends the STL Iterator class. The Vertex, Face, Cell, and Partition classes are simple proxies that utilize the globally unique integer *id* variables to fulfill user mesh queries by accessing appropriate fields in the *PartitionRaw* data structure. They are helper classes that hide the implementation details of *PartitionRaw* and provide a uniform access to its contents. Except for the *id* variables used to identify them, they hold no other data. *PartitionRaw* itself, is the distributed mesh data structure described in the next section. The *CELL_TYPE* is used by the *ReferenceMap*, *FEM* and *Quad* components to determine respectively, which reference element, which shape functions and which quadrature rule are appropriate.

These *ReferenceMap*, *FEM* and *Quad* modules abstract the notion of the a reference cell, finite element space, and a quadrature rule respectively. They provide the following interface:

```
template<int wdim>
class ReferenceMap {
        void reinit(const Cell<wdim>& cell);


        double jacobian(const Point<wdim>& quadNode);
        Point<wdim> map(const Point<wdim>& quadNode);
        ...
};
```

```
template<int wdim>
class FEM {
        FEM(enum FEM_TYPE, const Quad& quad);

        void reinit(const Cell<wdim>& cell);

        double evalShape(int shapeIndex,
                         const Point<wdim>& quadNode);
        ...
};
```

```
template<int wdim>
class Quad {
        Quad(enum QUAD_TYPE);

        void reinit(const Cell<wdim>& cell);

        double weight(int quadNodeIndex);
        Point<wdim> node(int quadNodeIndex);
        ...
};
```

The enumerated types *FEM_TYPE* and *QUAD_TYPE* are used to specify the type of the finite element space and the degree of precision of the quadrature rule respectively.

The *Assembler* and *LinearSystem* components define functions that need to be provided by the user:

```
template<int wdim>
class Assembler {
        Partition<wdim>* partition;

        ReferenceMapper<wdim>* refMap;
        FEM<wdim>* fem;
        Quad<wdim>* quad;

        virtual void assemble() = 0;
        ...
};
```

```
template<int wdim>
class LinearSystem {
        Partition<wdim>* partition;

        virtual void define() = 0;
        virtual void solve() = 0;

        ...
};
```

The *assemble* function assembles the linear system corresponding to the local partition, which is defined by the *define* function and *solve* specifies how to solve it.

To satisfy the last requirement, we introduced the *Solver* component into our framework. It resides on the same rank as the *Mesh* component. Its responsibility is to coordinate the process of determination of the local solutions by *LinearSystems* as well as the exchange and update phase in the domain decomposition algorithm. It also provides a natural interface which can be conveniently used by an iterative method encapsulated in the *Iterator* component.

```
template<int wdim, class T>
class Solver {
        Mesh<wdim>* mesh;

        vector<T> requestSolution(int partitionRank);
        virtual vector<T> operator*(const vector<T>& x) = 0;
        ...
};
```

The *requestSolution* contacts the *LinearSystem* with rank *partitionRank* and obtains a solution of the corresponding local problem. That information is then used by a user provided function *operator** which encapsulates the exchange and update of interface data in the domain decomposition method.

The *Iterator* component can now make use of the Solver in the following way:

```
Solver A;
X = A*x; // to calculate the action of the iteration operator
```

6.2 Distributed Mesh Data Structure

We should mention that the implementation of highly scalable, distributed meshes is very challenging. For example, deal.ii library mesh data structure is build on top of p4est [26] library which takes care of mesh partitioning, load balancing, parallel neighborhood context, etc. The lack of time and resources did not permit us to use a similar approach. We have therefore opted for a simpler parallel mesh representation.

However, since the access to the mesh data structure is moderated by the *Mesh Proxy* object, the future improvements to the *Mesh* component could be carried out without any changes to the existing code.

6.2.1 Compact Array-Based Mesh Data Structure

Since massively parallel simulations cannot afford to replicate the entire mesh on every cluster node, a robust distributed mesh data structure is necessary for the development of efficient domain decomposition algorithms. We have decided to adapt the compact, array-based mesh data structure [1] for our particular application. This structure is efficient in both time and space. The mesh entities and their neighborhood information can be queried without performing global search while requiring a minimal amount of storage. This characteristic is very important for an efficient determination of the cross-point vertices. The classic mesh representations for finite element analysis bases on element connectivity [44] do not support fast neighborhood queries. For example, finding all mesh triangles incident on a given vertex is linear in the total number of triangles. This is impractical. In order to implement automatic partitioning and apply the cross-point technique to the resulting nodal finite element system, we need this particular operation to perform very quickly, and the mesh data structure described here can achieve this in $O(1)$ time.

This data structure is based on *half-edge* mesh representation. For the sake of simplicity we assume that we are working with a 2D mesh composed of triangles. Extensions to higher dimensions are trivial [1]. In this representation, each triangle in addition to having a unique ID, is encoded as a linear sequence of *half-edges*. A half edge represents a triangle edge. However, unlike a regular edge, it is associated with only a single triangle. Its *twin half-edge* is the *half-edge* corresponding to the same mesh edge, contained in the neighboring triangle. A *half-edge* is represented by a tuple $< T, i >$, where T is the ID of the containing triangle and i is the index of the corresponding edge within that triangle. The following tables are used by the mesh data structure:

- **VC**: map each vertex ID to the vertex coordinates.
- **EC**: map each triangle ID to the IDs of its vertices.
- **V2e**: map each vertex to the ID of an incident half-edge originated from the vertex; map a border vertex to a border half-edge; map an interface vertex to a interface half-edge.
- **E2e**: map each non-border half-edge to the ID of its twin half-edge.
- **B2e**: map each border half-edge to the ID of its twin non-border half-edge.
- **Vg2l**: map global vertex ID to a local vertex ID within the containing partition.

The tables **V2e**, **E2e** and **B2e** are fundamental. Table **V2e** gives access to the local neighborhood of a vertex. By looking up an incident edge in **V2e** we can determine all half-edges incident on the vertex by consulting the **E2e** table. By looking at the the resulting sequence of half-edges we can determine which triangles they are contained in, whether they lie on a physical border or a partition interface by looking up the **B2e** table. A cross-point is a vertex which belongs to more than two partitions. If the above procedure, which takes a constant time for each vertex, finds incident half-edges that happen to lie on two different interfaces, then the vertex is a marked as possible cross-point. Every partition reports the set of these points to a single process on rank 0 whose responsibility is to hold the global *Mesh* component described in the previous section. The process then assigns the unique cross-point IDs to these vertices and communicates that information back to the participating partitions.

Mesh queries related to the assembly of the finite element system are also easily supported. In fact, what is needed are the triangle iterators which can be built from the **EC** table, and the interface and boundary edge iterators which can be implemented on top of the **B2e** table. Of course, the geometric information, ie, the vertex coordinates can be easily looked up in the **VC** table.

The mesh can be generated using any mesh generation software. We have used GMSH [42]. The output of GMSH is an element connectivity list with boundary information attached to the mesh entities. The element connectivity list can be fed into an automatic mesh partitioner such as METIS [50] producing an additional partition information attached to the mesh entities. The whole output then needs to be wrangled to populate the entries of the above tables. That is possible since the two mesh representations are equivalent [1].

Since the implementation of this mesh representation is hidden from the user implementing the domain decomposition algorithm, it can be easily exchanged for a more advanced data structure. The approach to mesh handling presented in this chapter suffers from the problem of scalability. The mesh needs to be generated and partitioned on a single node. This limits the possible mesh size. The more scalable solution would be to use local mesh refinements and recursive mesh splitting with load balancing. Such functionality is provided by the p4est library [26] and could potentially be used in our future work to improve the code to handle very large problems very efficiently without running into problems posed by local memory limitations.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

In this dissertation we have proposed new numerical approaches to the solution of the scattering problem. In Chapters 3 and 4 we have concentrated on the solution of the scattering problem truncated using the perfectly matched layer technique. We have introduced a new, zero frequency limit preconditioner to improve the convergence rate of Krylov subspace iterative methods applied to the PML system (Chapter 3). Our preconditioner reduces the number of required GMRES iterations by a large factor, is sparse and very easy to assemble. In comparison with the other popular choices of preconditioners for these type of problems, it achieves greatly improved performance at a fraction of the computational cost. This is especially true when we compare it with the approximate inverse preconditioner which if very difficult to construct. The parameter space of the PML problem is very large and a more exhaustive numerical study is needed to obtain a better picture of the relative performance of the zero frequency limit preconditioner. We have only covered the cases where σ^*, γ and k are being varied, and in particular, we did were unable to replicate the results for values of k higher than π as those could not be handled by Matlab due to their size. Future work should concentrate on a much more further exploration of the parameter space, considerations of higher values of the wavenumber k which should include the computation of the spectra of the corresponding matrices which we were unable to obtain for $k = 2\pi$ and $k = 3\pi$. To the best of our knowledge, the behavior of the spectra in the complex place is crucial to the understanding of the improved performance of the new preconditioner and therefore, the determination of eigenvalues is particularly important.

In Chapter 4 we have applied a domain decomposition method to the perfectly matched layer problem. This is a novel technique whose development has been motivated by the fact that the PML problem can become too large for a single computer to handle. We have analyzed the resulting method applied to a simple problem of scattering from a plane wall and showed that the use of square-root transmission operator guarantees convergence. However, since the operator is non local, its use in practical numerical simulations is not efficient. The state of the art localization procedure based on rational approximation to the symbol of the operator cannot be used in a straightforward manner since the natural decomposition of the PML problem geometry contains cross-points. We have proposed an adaptation of the cross-point algorithm that doesn't suffer from this deficiency. Our numerical studies have shown that the resulting domain decomposition method that combines the cross-point technique with the Padé type localization of the square-root transmission operator yields an efficient algorithm. In the context of the PML problem, we have only considered two dimensional geometries, a further study should focus on examining the performance of the algorithm in three dimensions although we have no reason to assume that it would deteriorate. Also, our analysis indicates that the convergence rate of the algorithm depends on the PML parameters, the the choice of their optimal values should be investigated as well.

The second part of the thesis introduced a new family of transmission conditions for domain decomposition methods for the scattering problem. To the best of our knowledge, there are no well-posedness results pertaining to local problems appearing in the current state of the art, quasi-optimal DDM that utilize Padé type localization of the transmission operator. We have also observed that the size of the discrete local problem increases with the order of the approximating series N_p. The introduction of new transmission conditions in Chapter 5 has been motivated to remove these two main deficiencies of the current methods just described. The form of the improved transmission conditions allows to prove well-posedness of local problems by choosing α appropriately whereas the operators T_i can be chosen non local, to improve convergence without affecting the structure of the local problems. Our numerical studies have shown that in the context of the square-root transmission operator, choosing α and T_i appropriately leads to a very efficient, and robust algorithm, especially when applied to three dimensional problems. There, the reduction in computational time over the current DDMs is very pronounced. Future work should include investigation of the method as the order of the approximation N_p is varied. Our numerical results were obtained for the typical value $N_p = 8$ which is commonly used. Also, an analysis of the convergence of the new method should be considered in the case of GMRES iterations. Such work could help to answer the question as to how pick the operators T_i to improve convergence even further.

In Chapter 6 we have discussed an extensible, object oriented architecture that supports development of parallel domain decomposition algorithms where local problems are solved using the finite element method. While there exist quite an extensive literature on object oriented design of finite element codes, to the best of our knowledge, the design of domain decomposition algorithms based on the same principles have not been thoroughly studied. Making the framework extensible is essential as it is very difficult to precisely state the system requirement in advance. Those requirements pertain the query capabilities of the mesh object, finite element spaces used, the degree of precision of quadrature formulas or the type of an iterative method to solve the DDM system. In Chapter 6 we gave an example of a framework for DDM codes that are independent of the problem dimensionality as well as the low level details listed above. We have implemented the resulting architecture using modern C++ techniques with great success. In the future, to achieve greater scalability, the underlying mesh representation should be modified to handle refinement, partitioning and load balancing dynamically, at run time to tackle the largest problems found in the industry. This would constitute the first step to hiding the very difficult and cumbersome task of mesh partitioning from the user and making the domain decomposition algorithms more automatized.

APPENDIX A

FINITE ELEMENT METHOD FOR A MODEL HELMHOLTZ BOUNDARY VALUE PROBLEM

A.1 Model Helmholtz Boundary Value Problem

We consider the problem of acoustic scattering of a signal g from a surface Γ of a bounded obstacle $\Omega_- \subset \mathbb{R}^2$. A Dirichlet-to-Neumann map is used as an absorbing boundary condition on an artificial surface Σ (see figure A.1):

$$\begin{aligned} \Delta u + k^2 u &= 0 \quad \text{in} \quad \Omega \\ \partial_n u &= g \quad \text{on} \quad \Gamma \\ \partial_n u - \Lambda u &= 0 \quad \text{on} \quad \Sigma \end{aligned} \tag{A.1}$$

where n is the unit normal vector to Γ and Σ pointing away from the computational domain Ω.

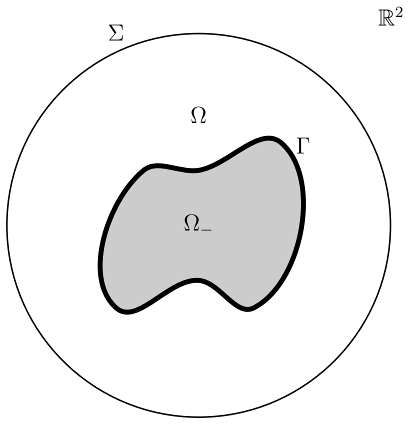


Figure A.1 Problem domain.

We take the operator Λ to be the Padé approximation to the Dirichlet-to-Neumann operator of the form:

$$\Lambda\, u = -ik \left(C_0 u + \sum_{l=1}^{N_p} A_l\, \partial_s \left(k_\epsilon^{-2} \partial_s \varphi_l \right) \right) \quad \text{on} \quad \Sigma \tag{A.2}$$

where $\varphi_l : l = 1, \dots, N_p$ are solutions to the auxiliary surface equations:

$$\varphi_l + B_l \, \partial_s \left(k_\epsilon^{-2} \partial_s \varphi_l \right) = u \quad \text{on} \quad \Sigma \tag{A.3}$$

with A_l, B_l and C_0 being the coefficients of Padé series of order N_p.

A.2 Weak Formulation

Let us denote by $L_2(\Omega)$ the space of square Lebesgue integrable functions over Ω. The multi-index α is an n-dimensional tuple:

$$\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)$$

of non-negative integers. The order of the index is $|\alpha| = \alpha_1 + \alpha_2 + \cdots + \alpha_n$ and it becomes the order of a derivative in the following notation:

$$\partial^\alpha = \partial_1^{\alpha_1} \partial_2^{\alpha_2} \dots \partial_n^{\alpha_n}$$

Definition A.2.1. Given an integer $m \geq 0$ we denote by $H^m(\Omega)$ the set of all functions $f \in L^2(\Omega)$ which poses weak derivatives $\partial^\alpha f$ for all $|\alpha| \leq m$.

Assuming $u,\ v \in H^1(\Omega)$ and $\varphi_l,\ v_l \in H^1(\Sigma)$, integration by parts gives a coupled variational formulation:

$$\int_\Omega \nabla u \cdot \nabla v - k^2 uv \, dA - ikC_0 \int_\Sigma uv \, ds + ik \sum_{l=1}^{N_p} A_l \int_\Sigma k_\epsilon^{-2} \partial_s \varphi_l \, \partial_s v \, ds = \int_\Gamma gv \, ds \tag{A.4}$$

$$\int_\Sigma \varphi_l v_l \, ds + B_l \int_\Sigma k_\epsilon^{-2} \partial_s \varphi_l \, \partial_s v_l \, ds - \int_\Sigma u v_l \, ds = 0 \quad \text{for} \quad l = 1, \dots, N_p \tag{A.5}$$

Let us denote by X_Ω, and X_Σ respectively, the Sobolev spaces $H^1(\Omega)$ and $H^1(\Sigma)$. We say that $u \in X_\Omega$ is a weak solution of (A.1) provided that (A.4)-(A.5) is satisfied for all test functions $v \in X_\Omega$ and all $v_l \in X_\Sigma$.

A.3 Galerkin Method

Let X_Ω^h be a finite N_Ω-dimensional subspace of X_Ω. The finite element method looks for a best approximation u^h to u in X_Ω^h. We introduce a basis for X_Ω^h comprised of functions ψ_i and write:

$$u^h = \sum_{i=1}^{N_\Omega} \alpha_i \psi_i \tag{A.6}$$

where the coefficients $\alpha_i \in \mathbb{C}^1$. Restriction of the basis ψ_i to Σ induces a basis for the N_Σ-dimensional subspace X_Σ^h of X_Σ containing functions $\omega_i : 1, \dots, N_\Sigma$. To be more precise, assume that the basis for X_Ω^h has been enumerated in such a way that $\{\psi_i\}_{i=1}^{N_\Sigma}$ are the only basis functions with a non zero trace on Σ. Then we define $\omega_i = \psi_i|_\Sigma$ for $i = 1, \dots, N_\Sigma$. The approximate solution $\varphi_l^h \in X_\Sigma^h$ to problem (A.5) may be written as:

$$\varphi_l^h = \sum_{i=1}^{N_\Sigma} \beta_{li} \omega_i \tag{A.7}$$

where the coefficients $\beta_{li} \in \mathbb{C}^1$. Since the variational equality (A.4)-(A.5) holds for all u^h, $v \in X_\Omega^h$, and all φ^h, $v_l \in X_\Sigma^h$ it evidently holds on the basis function $\{\psi_i\}_{i=1}^{N_\Omega}$, and $\{\omega_i\}_{i=1}^{N_\Sigma}$. Galerkin method consists of substituting the expansions for u^h and $\varphi_l^h : l = 1, \dots, N_\Sigma$ into (A.4)-(A.5) and evaluating it on every basis function which yields:

$$\left(\mathbf{S}^\Omega - k^2\mathbf{M}^\Omega - ikC_0\mathbf{M}^\Sigma\right)\mathbf{u} + ik\sum_{l=1}^{N_p} A_l \mathbf{S}^\Sigma_{k_\epsilon^{-2}} \boldsymbol{\varphi}_l = \mathbf{M}^\Gamma \mathbf{g} \tag{A.8}$$

$$-\mathbf{M}^\Sigma \mathbf{u} + \left(\mathbf{M}^\Sigma - B_l \mathbf{S}^\Sigma_{k_\epsilon^{-2}}\right)\boldsymbol{\varphi}_l = \mathbf{0} \quad l = 1, \dots, N_p \tag{A.9}$$

where $\mathbf{u}$ and $\boldsymbol{\varphi}_l$ are the vectors of coefficients with $[\mathbf{u}]_i = \alpha_i$ and $[\boldsymbol{\varphi}_l]_i = \beta_{li}$. The matrices $\mathbf{S}^\Omega$ and $\mathbf{M}^\Omega$ denote the stiffness and mass matrices associated with domain

Ω. That is:

$$\left[\mathbf{S}^{\Omega}\right]_{ij} = \int_{\Omega} \nabla\psi_i \cdot \nabla\psi_j \, dA \qquad \left[\mathbf{M}^{\Omega}\right]_{ij} = \int_{\Omega} \psi_i\psi_j \, dA \tag{A.10}$$

Similarly, matrices $\mathbf{S}^{\Sigma}$ and $\mathbf{M}^{\Sigma}$ are respectively the stiffness and mass matrices associated with the boundary Σ:

$$\left[\mathbf{S}^{\Sigma}\right]_{ij} = \int_{\Sigma} \partial_s\omega_i\partial_s\omega_j \, ds \qquad \left[\mathbf{M}^{\Sigma}\right]_{ij} = \int_{\Sigma} \omega_i\omega_j \, ds \tag{A.11}$$

Also, the generalized stiffness matrix $\mathbf{S}^{\Sigma}_{k_\epsilon^{-2}}$ corresponding to a surface function k_ϵ^{-2} over Σ is:

$$\mathbf{S}^{\Sigma}_{k_\epsilon^{-2}} = \int_{\Sigma} k_\epsilon^{-2}\partial_s\omega_i\partial_s\omega_j \, ds \tag{A.12}$$

We have abused the notation slightly in equations (A.8) - (A.9). For example, in equation (A.8) the matrices $\mathbf{S}^{\Omega}$ and $\mathbf{M}^{\Omega}$ are both N_Ω by N_Ω whereas matrix $\mathbf{S}^{\Sigma}_{k_\epsilon^{-2}}$ has size N_Σ by N_Σ. If we introduce a basis for $X^h_\Gamma = H^1(\Gamma)$ just as we did for the space X^h_Ω we would see that the dimensions of the right hand side are also inconsistent. What we haven't included in those equations for the sake of clarity are the various matrices that establish a mapping between bases of different subspaces. For instance, in order to make the equation (A.8) dimensionally correct we need to define a matrix $\mathbf{R}^{\Sigma}_{\Omega} : \mathbb{C}^{N_\Sigma} \to \mathbb{C}^{N_\Omega}$ such that:

$$\left[\mathbf{R}^{\Sigma}_{\Omega}\boldsymbol{\varphi}_l\right]_i = \begin{cases} \beta_{li} & \text{if} \quad \psi_i|_\Sigma \neq 0 \\ 0 & \text{otherwise} \end{cases} \qquad i = 1, \ldots, N_\Omega \tag{A.13}$$

Defining matrix $\mathbf{R}^{\Gamma}_{\Omega}$ in a similar fashion we would have written the equation (A.8) as follows:

$$\left(\mathbf{S}^{\Omega} - k^2\mathbf{M}^{\Omega} - ikC_0\mathbf{M}^{\Sigma}\right)\mathbf{u} + ik\sum_{l=1}^{N_p} A_l\mathbf{R}^{\Sigma}_{\Omega}\,\mathbf{S}^{\Sigma}_{k_\epsilon^{-2}}\boldsymbol{\varphi}_l = \mathbf{R}^{\Gamma}_{\Omega}\,\mathbf{M}^{\Gamma}\mathbf{g} \tag{A.14}$$

The action of a matrix $\mathbf{R}_{\Sigma}^{\Omega} : \mathbb{C}^{N_\Omega} \to \mathbb{C}^{N_\Sigma}$ is defined as:

$$\left[\mathbf{R}_{\Sigma}^{\Omega}\boldsymbol{\varphi}_l\right]_i = \alpha_j \quad \text{where} \quad j \quad \text{is such that} \quad \psi_j|_\Sigma = \omega_i \quad \text{for} \quad i = 1, \dots, N_\Sigma \tag{A.15}$$

Consequently the precise, dimensionaly correct form of (A.9) is:

$$-\mathbf{R}_{\Sigma}^{\Omega}\mathbf{M}^{\Sigma}\mathbf{u} + \left(\mathbf{M}^{\Sigma} - B_l \mathbf{S}_{k_\epsilon^{-2}}^{\Sigma}\right)\boldsymbol{\varphi}_l = \mathbf{0} \quad l = 1, \dots, N_p \tag{A.16}$$

In practice however, the matrices $\mathbf{R}$ are not explicitly constructed. The assembly of the system (A.17) is the most conveniently performed by considering a single block matrix at a time and distributing its entries accordingly to the mappings between the various bases described above.

$$\begin{bmatrix} \mathbf{S}^{\Omega} - k^2\mathbf{M}^{\Omega} - ikC_0\mathbf{M}^{\Sigma} & ikA_1\mathbf{R}_{\Omega}^{\Sigma}\mathbf{S}_{k_\epsilon^{-2}}^{\Sigma} & \dots & ikA_{N_p}\mathbf{R}_{\Omega}^{\Sigma}\mathbf{S}_{k_\epsilon^{-2}}^{\Sigma} \\ -\mathbf{R}_{\Sigma}^{\Omega}\mathbf{M}^{\Sigma} & \mathbf{M}^{\Sigma} - B_1\mathbf{S}_{k_\epsilon^{-2}}^{\Sigma} & \mathbf{0} & \dots \\ \vdots & \ddots & \ddots & \vdots \\ -\mathbf{R}_{\Sigma}^{\Omega}\mathbf{M}^{\Sigma} & \mathbf{0} & \dots & \mathbf{M}^{\Sigma} - B_{N_p}\mathbf{S}_{k_\epsilon^{-2}}^{\Sigma} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \boldsymbol{\varphi}_1 \\ \vdots \\ \boldsymbol{\varphi}_{N_p} \end{bmatrix} = \\ = \begin{bmatrix} \mathbf{R}_{\Omega}^{\Gamma}\mathbf{M}^{\Gamma}\mathbf{g} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} \tag{A.17}$$

A.4 Finite Element Functions

A convenient choice of X_Ω^h corresponds to the space of piecewise polynomial functions. To define such functions we partition the computational domain Ω into finitely many regions referred to as *elements*. For planar problems, they can be triangles or quadrilaterals. For three-dimensional problems, one can use tetrahedra, cubes, prisms, rectangular parallelepipeds, etc. For the solution of our model problem we will employ a triangular mesh over Ω denoted by T_Ω^h. We assume that triangulation

T^h_Ω is a collection of N_T triangles $T_i : i = 1, \ldots, N_T$ that is admissible in the following sense:

- if $T_i \cap T_j$ consists of exactly one point, then it is a common vertex of T_i and T_j,
- if for $i \neq j$, $T_i \cap T_j$ consists of more than one point, then $T_i \cap T_j$ is a common edge of T_i and T_j.

If X^h_Ω is the space of piecewise polynomials of degree $\leq d$ then the restriction of every function in that space to a mesh triangle is a polynomial of degree $\leq d$. These functions are called *finite elements.* The finite element space is said to be C^k-element provided it is contained in $C^k(\Omega)$. The degree of smoothness of the finite element space necessary for the Galerkin method to achieve the best approximation is stated in the following therem [24, chapter 5]:

Theorem A.4.1. *Let $k \geq 1$ and suppose Ω is bounded. Then a piecewise infinitely differentiable function $v : \overline{\Omega} \to \mathbb{R}$ belongs to $H^k(\Omega)$ if and only if $v \in C^{k-1}(\overline{\Omega})$.*

A.5 Nodal Basis

Definition A.5.1. Suppose that for a given finite element space there exists a set of points which uniquely determines any function in the space by its values at the points. Then the set of functions in the space which take a value of one at precisely one of the points and zero everywhere else forms a basis for the space, called the *nodal basis.*

For the polynomial finite element spaces, the existence of the nodal basis is guaranteed by the following theorem [24, chapter 5]:

Theorem A.5.2. *Let $d \geq 0$. Give a triangle T, suppose $z_1, z_2, \ldots, z_{N_f}$ are the $s = 1 + 2 + \cdots + (d+1)$ points in T which lie on $d+1$ lines parallel to one side of a triangle. Then for every $f \in C(T)$, there is a unique polynomial ψ of degree $\leq d$*

satisfying the interpolation conditions:

$$\psi(z_j) = f(z_j) \tag{A.18}$$

Example A.5.3. A Nodal Basis for C^0-elements. For our model problem, and in fact for many second order problems, theorem (A.4.1) guarantees that globally continuous finite elements are sufficient to approximate a solution in the $H^1(\Omega)$ space. In comparison with higher order elements, the construction of the nodal basis for C^0 finite element space is considerably less involved. As an example we construct a nodal basis for the finite element space X^h_Ω consisting of piecewise polynomials of degree $\leq d$ for the solution of our model problem. To this end, in each triangle $T \in T^h_\Omega$ we place $s := (d+1)(d+2)/2$ points along the $d+1$ lines parallel to one side of the triangle so that there are $d+1$ points on each edge (see figure A.2).

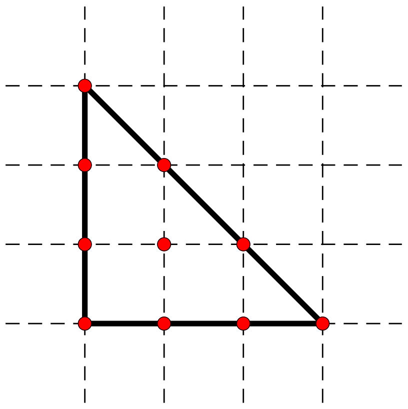


Figure A.2 Nodes of the nodal basis for cubic triangular element (red dots).

By theorem (A.5.2) s polynomials $\psi_i : i = 1, \ldots, s$ of degree d in two variables can be found such that $\psi_i(z_j) = \delta_{ij}$. The restriction of any such polynomial to an edge is a polynomial of degree d in one variable. Now given an edge, the two polynomials on either side must interpolate the same values at the $d+1$ points on that edge, and thus must reduce to the same one-dimensional polynomial. This ensures that the finite elements are globally continuous.

One advantage of using piecewise polynomial finite element space as the approximating subspace in the Galerking method is the ease with which polynomials can be evaluated, differentiated and integrated. The second reason is that when the standard nodal basis is used as described above, the resulting block matrices in (A.17) are all sparse.

From the computational point of view however, the determination of nodal basis on each triangle can be quite expensive as it requires inversion of s dense matrices each with dimension s by s associated with the interpolation condition $\psi_i(z_j) = \delta_{ij} : i, j = 1, \ldots, s$. Since polynomials are invariant under an affine linear transformation it is more efficient to define the nodal basis over a reference triangle T_R with vertices at $(0, 0)$, $(0, 1)$ and $(1, 0)$. The nodal basis functions over a reference element are called *shape functions* and are easy to determine. The affine linear transformation γ_T that takes T_R into an arbitrary triangle T also transforms a shape function $\phi_i : i = 1, \ldots, s$ into a nodal basis function ψ_i over T. The transformation γ_T is easy to compute. The advantage of this approach is twofold. Firstly, the shape functions can be precomputed ahead of time. Thus the work involved in determination of ψ_i over a triangle T is equal to the work involved in an application of γ_T to ϕ_i. Secondly, the assembly of the matrix blocks of (A.17) requires the use of numerical quadrature in the most general case in which the related functionals contain non-constant coefficient terms. Since the quadrature formulas are most often specified on reference simplexes such as T_R, the use of shape functions is well justified.

A.6 Linear Shape Functions over a Reference Triangle

The reference triangle T_R is the triangle with vertices $\mathbf{u}_0 = (0, 0)$, $\mathbf{u}_1 = (0, 1)$ and $\mathbf{u}_2 = (1, 0)$. Let us denote an arbitrary point in T_R by $\mathbf{u} = (s, t)$. The reference triangle T_R is mapped to a triangle T with vertices $\mathbf{z}_0 = (x_0, y_0)$, $\mathbf{z}_1 = (x_1, y_1)$ and

$\mathbf{z}_2 = (x_2, y_2)$ by the following transformation:

$$\begin{aligned} x &= x_0 + (x_1 - x_0)s + (x_2 - x_0)t \\ y &= y_0 + (y_1 - y_0)s + (y_2 - y_0)t \end{aligned} \tag{A.19}$$

which sends $\mathbf{u}_0$ to $\mathbf{z}_0$, $\mathbf{u}_1$ to $\mathbf{z}_1$ and $\mathbf{u}_2$ to $\mathbf{z}_2$ and where $\mathbf{z} = (x, y)$ is a corresponding point in T. Equation (A.19) in matix form becomes:

$$\mathbf{z} = \mathbf{z}_0 + \mathbf{J}\mathbf{u} \tag{A.20}$$

where the matrix $\mathbf{J}$ is:

$$\mathbf{J} = \begin{bmatrix} x_1 - x_0 & x_2 - x_0 \\ y_1 - y_0 & y_2 - y_0 \end{bmatrix}$$

Suppose that we are interested in piecewise linear finite element space. That is, the nodal basis reduces to a polynomial of degree one on every triangle T. The reference mapping $\gamma_T : \mathbf{u} \in T_R \to \mathbf{z} \in T$ given by (A.20) can now be used to define the linear shape functions $\phi_i : i = 0, 1, 2$ that satisfy: $\phi_i(\mathbf{u}) = \psi_{k(i)}(\gamma_T(\mathbf{u}))$ where $k(i) \in \{1, 2, \ldots, N_f\}$ is the global index of the k^{th} vertex of T in the mesh. If such functions exist, the interpolation condition on the nodal basis yields: $\phi_i(\mathbf{u}_j) = \psi_{k(i)}(\gamma_T(\mathbf{u}_j)) = \delta_{ij}$. Since γ_T is an affine linear transformation and since $\psi_{k(i)}$ are linear, the shape functions must also be linear over T_R. These two conditions on ϕ_j are easy to fulfill. In fact:

$$\begin{aligned} \phi_0 &= 1 - s - t \\ \phi_1 &= s \\ \phi_2 &= t \end{aligned} \tag{A.21}$$

are just the functions we are looking for. The relationship between $\nabla_{(s,t)}\phi_i$ and $\nabla_{(x,y)}\psi_{k(i)}$ follows from the chain rule:

$$\nabla_{(x,y)}\psi_{k(i)} = \mathbf{J}^{-T}\nabla_{(s,t)}\phi_i \tag{A.22}$$

A.7 Assembly of the Element Matrices

Assembly of the linear system (A.17) refers to the process in which its entries are populated according to equatiosn (A.17), (A.10), (A.12) and so forth. This procedure involves evaluation of integrals over Ω as well as its boundary composed of Γ and Σ. We will instead consider integrals over triangular patches and edges induced by the mesh, that cover the domain and its boundary. Thus for example, the entries of the stiffness matrix associated with the domain Ω are approximated by:

$$\left[\mathbf{S}^{\Omega}\right]_{ij} \approx \sum_{T\in T^h_{\Omega}} \int_T \nabla\psi_i \cdot \nabla\psi_j \, dA \tag{A.23}$$

with equality holding only in the case of Ω begin a polygonal domain. In a general case when Ω is not polygonal, the above entries contain truncation error due geometric approximation of Ω by T^h_{Ω}.

The integrals over triangles T in (A.23) are easy to evaluate with the use of Gaussian quadrature formulas for a reference triangle. Suppose that such formula consists of $N_G^{T_R}$ points $\{(s_i,t_i)\}_{i=1}^{N_G^{T_R}}$ and weights $\{w_i\}_{i=1}^{N_G^{T_R}}$ then we may approximate:

$$\begin{aligned}\int_T \nabla\psi_i \cdot \nabla\psi_j \, dA &= \int_{T_R} \mathbf{J}^{-T}\nabla_{(s,t)}\phi_i \cdot \mathbf{J}^{-T}\nabla_{(s,t)}\phi_j \,|\mathbf{J}|\, dA \\ &\approx \sum_{i=1}^{N_G^{T_R}} w_i \, \mathbf{J}^{-T}\nabla_{(s,t)}\phi_i(s_i,t_i) \cdot \mathbf{J}^{-T}\nabla_{(s,t)}\phi_j(s_i,t_i)\,|\mathbf{J}|\end{aligned} \tag{A.24}$$

where $|\mathbf{J}|$ is the Jacobian determinant of the transformation γ_T.

The process of assembly of the surface stiffness and mass matrices is analogous. Line integrals are approximated by integrals along straight line segments, which are the edges of the triangles in the mesh. For example:

$$\left[\mathbf{S}^{\Sigma}\right]_{ij} = \int_{\Sigma} \partial_s \omega_i \partial_s \omega_j \, ds \approx \sum_{e \in T_{\Sigma}^h} \int_e \partial_s \omega_i \partial_s \omega_j \, ds \tag{A.25}$$

Every edge e can be mapped onto an interval $[-1, 1]$ on which standard single variable Gaussian quadrature formulas are defined. The parametric representation of and edge e with endpoints P_0 and P_1 is simply:

$$e(t) = \frac{P_0 + P_1}{2} + \frac{P_1 - P_0}{2} t \quad \text{for} \quad t \in [-1, 1] \tag{A.26}$$

Consequently:

$$\int_e \partial_s \omega_i \partial_s \omega_j \, ds = \int_{-1}^{1} \partial_t \zeta_i \partial_t \zeta_j \left| \frac{de}{dt} \right| \, dt \tag{A.27}$$

where $\zeta_i(t) = \omega_i(e(t))$. In the case of linear piecewise polynomials whose restriction to a triangle edge is a linear function the edge shape functions are respectively:

$$\zeta_0(t) = \frac{1-t}{2} \qquad \zeta_1(t) = \frac{1+t}{2} \tag{A.28}$$

Given an $N_G^{e_R}$ Gaussian quadrature formula on $e_R = [-1, 1]$ with points $\{t_i\}_{i=1}^{N_G^{e_R}}$ and weights $\{w_i\}_{i=1}^{N_G^{e_R}}$ we approximate the above by:

$$\int_{-1}^{1} \partial_t \zeta_i \partial_t \zeta_j \left| \frac{de}{dt} \right| \, dt \approx \sum_{i=1}^{N_G^{e_R}} w_i \, \partial_t \zeta_i(t_i) \partial_t \zeta_j(t_i) \left| \frac{de}{dt} \right| \tag{A.29}$$

APPENDIX B

PERFECTLY MATCHED LAYER TECHNIQUE

B.1 Derivation of PML Equations for the Scattering Problem

Derivation of the perfectly matched layer equations for the acoustic scattering problem is now well know. Detailed presentations of the technique can be found in [48] and [30]. Here, we only outline the procedure to provide context for our discussion and indicate the major features of the method.

We begin with the linearized equations of motion of an inviscid fluid (1.6) - (1.7) derived in section (1.2). To recall, they take the form:

$$\frac{\partial \tilde{\rho}}{\partial t} + \rho_0 \nabla \cdot \tilde{\mathbf{v}} = 0 \tag{B.1}$$

$$\frac{\partial \tilde{\mathbf{v}}}{\partial t} = -\frac{1}{\rho_0} \nabla \tilde{p} \tag{B.2}$$

Assume that the x and y components of the velocity vector $\tilde{\mathbf{v}}$ are respectively $\tilde{v}_1$ and $\tilde{v}_2$. Now, following Ihlenburg [48] we rewrite the mass conservation equation (B.1) as follows:

$$\frac{\partial \tilde{\rho}_1}{\partial t} + \rho_0 \frac{\partial \tilde{v}_1}{\partial x} = 0 \tag{B.3}$$

$$\frac{\partial \tilde{\rho}_2}{\partial t} + \rho_0 \frac{\partial \tilde{v}_2}{\partial y} = 0 \tag{B.4}$$

where we must require that $\tilde{\rho}$ assumes a formal decomposition $\tilde{\rho} = \tilde{\rho}_1 + \tilde{\rho}_2$. Quantities $\tilde{\rho}_1$ and $\tilde{\rho}_2$ carry no physical significance and are introduced only to facilitate the derivation of the PML equations. We also write the leading order momentum equation (B.2) in the component form:

$$\frac{\partial \tilde{v}_1}{\partial t} = -\frac{1}{\rho_0} \frac{\partial p}{\partial x} \tag{B.5}$$

$$\frac{\partial \tilde{v}_2}{\partial t} = -\frac{1}{\rho_0} \frac{\partial p}{\partial y} \tag{B.6}$$

We recall that the material law (1.9) relating $\tilde{\rho}$ and $\tilde{p}$ assumes the form:

$$\tilde{\rho} = c^2 \tilde{p} \tag{B.7}$$

We assume that the time-harmonic acoustic waves propagate in free space. We place the layer interface at $x = 0$. The idea of PML is to modify equations in the layer corresponding to the region of space with $x \geq 0$ in such a way as to obtain exponentially decaying solutions. This behavior is achieved by introducing an absorption term in the equations that contain derivative orthogonal to the PML interface. In our case this is simply the x-direction. The modified equations (B.3) and (B.5) read as follows:

$$\frac{\partial \tilde{\rho}_1}{\partial t} + \sigma_x(x)\tilde{\rho}_1 = -\rho_0 \frac{\partial \tilde{v}_1}{\partial x} \tag{B.8}$$

and

$$\frac{\partial \tilde{v}_1}{\partial t} + \sigma_x(x)\tilde{v}_1 = -\frac{1}{\rho_0}\frac{\partial p}{\partial x} \tag{B.9}$$

Since we are considering time-harmonic waves, the solutions to the equations above obtained by the method of integrating factors are of the form $A\, e^{-\sigma_x(x)t}$. Hence, in order to obtain decaying solutions inside of the layer we must require that $\sigma_x(x) \geq 0$ for $x \geq 0$. Furthermore, we require that $\sigma_x \equiv 0$ for $x < 0$ and $\sigma_x(x) \in C^1(\mathbb{R}^1)$.

To deduce the modified time-harmonic equation for $\tilde{p}$ we replace time derivatives in equations (B.8), (B.4), (B.9) and (B.6) with $-i\omega$ to obtain respectively:

$$\tilde{\rho}_1 = \frac{\rho_0}{\sigma_x - i\omega}\frac{\partial \tilde{v}_1}{\partial x} \tag{B.10}$$

$$\tilde{\rho}_2 = \frac{\rho_0}{i\omega}\frac{\partial \tilde{v}_2}{\partial y} \tag{B.11}$$

$$\tilde{v}_1 = -\frac{1}{\rho_0(\sigma_x - i\omega)}\frac{\partial \tilde{p}}{\partial x} \tag{B.12}$$

$$\tilde{v}_2 = -\frac{1}{i\omega\rho_0}\frac{\partial \tilde{p}}{\partial y} \tag{B.13}$$

Deriving (B.12) and (B.13) with time, inserting the result into (B.10) and (B.11) and finally using the material relation (B.7) gives us after some simplification:

$$\frac{i\omega}{\sigma_x - i\omega}\frac{\partial}{\partial x}\left(\frac{i\omega}{\sigma_x - i\omega}\frac{\partial \tilde{p}}{\partial x}\right) + \frac{\partial^2 \tilde{p}}{\partial y^2} + k^2\tilde{p} = 0 \tag{B.14}$$

We note that the above equation correctly reduces to a Helmholtz equation in the computational domain corresponding to $x \leq 0$. Additionally, since there is no jump in the material properties of the acoustic medium at the PML interface $x = 0$, the interface is completely transparent, that is, it produces no spurious reflections.

For computational applications, the infinite PML region needs to be truncated with a boundary at $x = d$ where the thickness of the layer $d \geq 0$. Correspondingly, a condition on $\tilde{p}$ needs to be specified on the terminating boundary. The choice of the boundary condition seems to have little effect on the performance of the PML [64]. A popular choice is a homogeneous Dirichlet type boundary condition. In that case the the artificial boundary gives raise to reflected waves that propagate through the layer back into the computational domain. However, due to the exponential decay, the amplitude of the spurious waves is negligible and in fact can be controlled by choosing the layer thickness parameter d sufficiently large. A simple analysis of this behavior is presented in [18].

An extension of the above derivation to the problem of acoustic scattering from a bounded obstacle in 2D is straightforward. As an example, in our model problem (2.1) we introduce the PML in both x and y directions to enclose the obstacle. Figure (B.1) shows the layer denoted by Ω_A placed at $x = \pm a$, $y = \pm b$ of thickness d. The layer is now supposed to absorb waves traveling both in x and y directions. Since the PML interface Γ_I is parallel to the x and y axes, in addition to including an absorption term into the decomposed equations of motion containing only the x-derivatives, we also introduce a similar term in the other two equations containing the y-derivatives.

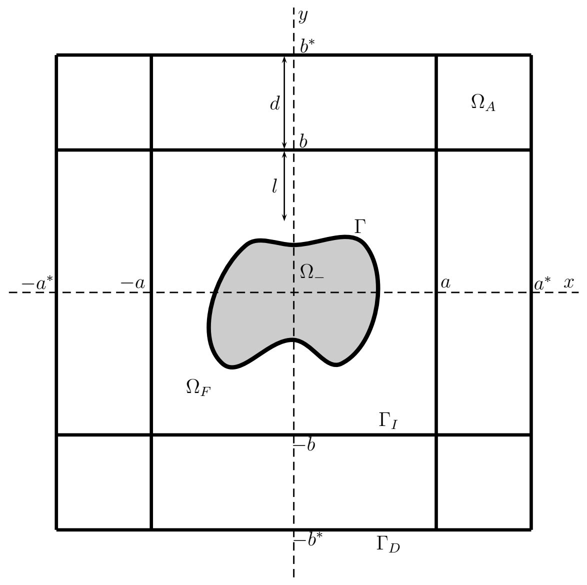


Figure B.1 Perfectly matched layer Ω_A surrounding the obstacle and the near field region Ω_F.

More precisely, in addition to equations (B.8) and (B.9) we also have the following two equations:

$$\frac{\partial \tilde{\rho}_2}{\partial t} + \sigma_y(y)\tilde{\rho}_1 = -\rho_0 \frac{\partial \tilde{v}_1}{\partial y} \tag{B.15}$$

$$\frac{\partial \tilde{v}_2}{\partial t} + \sigma_y(y)\tilde{v}_2 = -\frac{1}{\rho_0}\frac{\partial p}{\partial y} \tag{B.16}$$

Of course, definition of the absorbing functions σ_x and σ_y needs to be modified accordingly to our PML geometry. We recall that for $|x| > a$ and $|y| > b$ we require that $\sigma_x(x) \geq 0$ and $\sigma_y(y) \geq 0$. Furthermore, we also require that $\sigma_x(x) \equiv 0$, $\sigma_y(y) \equiv 0$ for $|x| \leq 0$, $|y| \leq 0$ respectively and that both functions belong to $C^1(\mathbb{R})$.

Assuming time-harmonic waves, derivation of the modified pressure equation proceeds in exactly the same way as for the case of a single PML strip acting in the x-direction. This equation takes the form:

$$\frac{i\omega}{\sigma_x - i\omega}\frac{\partial}{\partial x}\left(\frac{i\omega}{\sigma_x - i\omega}\frac{\partial \tilde{p}}{\partial x}\right) + \frac{i\omega}{\sigma_y - i\omega}\frac{\partial}{\partial y}\left(\frac{i\omega}{\sigma_y - i\omega}\frac{\partial \tilde{p}}{\partial y}\right) + \frac{\partial^2 \tilde{p}}{\partial y^2} + k^2\tilde{p} = 0 \tag{B.17}$$

To show that the amplitude of the velocity fluctuations satisfy the same modified equation in the PML region we proceed by first introducing a coordinate transformation:

$$\hat{x} = x + \frac{i}{\omega}\int_0^x \sigma_x(\xi)\,d\xi \qquad \text{and} \qquad \hat{y} = y + \frac{i}{\omega}\int_0^y \sigma_y(\eta)\,d\eta \tag{B.18}$$

Application of chain rule shows that the derivatives transform correspondingly into:

$$\frac{\partial}{\partial x} = -\frac{\sigma_x - i\omega}{i\omega}\frac{\partial}{\partial \hat{x}} \qquad \text{and} \qquad \frac{\partial}{\partial y} = -\frac{\sigma_y - i\omega}{i\omega}\frac{\partial}{\partial \hat{y}} \tag{B.19}$$

Therefore, in the $\hat{x}$, $\hat{y}$ coordinates the equation (B.17) simplifies to the original Helmholtz equation:

$$\frac{\partial^2 \tilde{p}}{\partial \hat{x}^2} + \frac{\partial^2 \tilde{p}}{\partial \hat{y}^2} + k^2 \tilde{p} = 0 \tag{B.20}$$

The above equation in time domain corresponds to a wave equation for the modified pressure function $\tilde{p}$. This is a function of $\hat{x}$, $\hat{y}$ and t with temporal dependence of the form $e^{-i\omega t}$. Consequently, derivation of the Helmholtz equation for the amplitude of the complex velocity potential follows the development of section (1.2) where we substitute $\hat{x}$ and $\hat{y}$ for the original spatial variables x and y respectively. Terminating the layer equation with a homogeneous Dirichlet boundary condition, and denoting the amplitudes of the scattered fields in the computational and PML regions by u_F^s and u_A^s respectively results in a system:

$$\Delta u_F^s + k^2 u_F^s = 0 \quad \text{in} \quad \Omega_F \tag{B.21}$$

$$\gamma_x^{-1}\partial_x\left(\gamma_x^{-1}\partial_x u_A^s\right) + \gamma_y^{-1}\partial_y\left(\gamma_y^{-1}\partial_y u_A^s\right) + k^2 u_A^s = 0 \quad \text{in} \quad \Omega_A \tag{B.22}$$

$$\partial_\nu u_F^s = -\partial_\nu u^i \quad \text{on} \quad \Gamma \tag{B.23}$$

$$u_F^s = u_A^s \quad \text{on} \quad \Gamma_I \tag{B.24}$$

$$\partial_\nu u_F^s = -\partial_\nu u_A^s \quad \text{on} \quad \Gamma_I \tag{B.25}$$

$$u_A^s = 0 \quad \text{on} \quad \Gamma_D \tag{B.26}$$

where

$$\gamma_x(x) = \begin{cases} 1 & x < a, \\ 1 + \frac{i}{k}\sigma_x(|x|) & a \le |x| < a^* \end{cases} \qquad \text{and} \qquad \gamma_y(x) = \begin{cases} 1 & y < b, \\ 1 + \frac{i}{k}\sigma_y(|y|) & b \le |y| < b^* \end{cases} \tag{B.27}$$

where ν is the vector normal to Γ_I pointing away from Ω_F. For simplicity, we take $c = 1$ in the above.

B.2 Variational Formulation

Let $\Omega = \Omega_F \cup \Omega_A$ denote the computational domain. The appropriate space in which to look for the solution to our Helmholtz problem is the subspace of the Sobolev space $H^1(\Omega)$ consisting of functions which vanish on the boundary Γ_D [48]. We will denote it by X. That is, $X = \{v \in H^1(\Omega) : v|_{\Gamma_D} = 0\}$. Now, let $v \in X$ be a test function. An application of divergence theorem over Ω_F gives:

$$\int_{\Omega_F} \nabla u_F \cdot \nabla v \, dx\, dy - k^2 \int_{\Omega_F} u_F v \, dx\, dy - \int_{\Gamma} gv \, ds = \int_{\Gamma_I} v \frac{\partial u_F}{\partial \nu_F} \, ds \tag{B.28}$$

With little more work and noting that $\gamma_x = \gamma_y = 1$ on Γ_I a similar expression can be derived for Ω_A:

$$\int_{\Omega_A} \gamma_y \gamma_x^{-1} \partial_x u_A \, \partial_x v \, dA + \int_{\Omega_A} \gamma_x \gamma_y^{-1} \partial_y u_A \, \partial_x v \, dA - k^2 \int_{\Omega_A} \gamma_x \gamma_y u_A v \, dA = \int_{\Gamma_I} v \, \partial_{\nu_A} u_A \, ds \tag{B.29}$$

Using the interface condition $\partial_{\nu_F} u_F = -\partial_{\nu_A} u_A$ on Γ_I we obtain:

$$\begin{aligned} \int_{\Omega_F} \nabla u_F \cdot \nabla v \, dA - k^2 \int_{\Omega_F} u_F v \, dA + \int_{\Omega_A} \gamma_y \gamma_x^{-1} \partial_x u_A \, \partial_x v \, dA + \\ + \int_{\Omega_A} \gamma_x \gamma_y^{-1} \partial_y u_A \, \partial_y v \, dA - k^2 \int_{\Omega_A} \gamma_x \gamma_y u_A v \, dA = \int_{\Gamma} gv \, ds \end{aligned} \tag{B.30}$$

Let us rewrite equation (B.30) as:

$$a_F(u_F, v) + a_A(u_A, v) = l(v) \tag{B.31}$$

where:

$$
\begin{aligned}
a_A(u,v) &= \int_{\Omega_F} \nabla u \cdot \nabla v \, dx\, dy - k^2 \int_{\Omega_F} uv \, dx\, dy \\
a_F(u,v) &= \int_{\Omega_A} \gamma_y \gamma_x^{-1} \partial_x u \, \partial_x v \, dA + \int_{\Omega_A} \gamma_x \gamma_y^{-1} \partial_y u \, \partial_y v \, dA - k^2 \int_{\Omega_A} \gamma_x \gamma_y uv \, dA \\
l(v) &= \int_\Gamma gv \, ds
\end{aligned}
\tag{B.32}
$$

Let us also define X_F and X_A to be the subspaces of X containing functions that vanish on Ω_A and Ω_F respectively. Then the variational problem is to find $u_F \in X_F$ and $u_A \in X_A$ such that $a_F(u_F, v) + a_A(u_A, v) = l(v)$ where $u_F|_{\Gamma_I} = u_A|_{\Gamma_I}$ for all $v \in X$.

B.3 Nodal Finite Element Method

We approximate the solution to (B.31) using triangular finite elements. To this end we introduce triangulation T^h of the computational domain Ω where parameter h denotes the mesh size. We assume that T^h induces conforming triangulations of Ω_F and Ω_A which we denote by Ω_F^h and Ω_A^h respectively. We let X^h to be the finite dimensional subspace of X consisting of piecewise linear polynomials. More precisely, $X^h = \{v^h \in X : v^h|_T \in \mathbb{P}^1, \forall T \in T^h\}$. For $u^h, v^h \in X^h$ we define the discrete version of the functionals in (B.33) as follows:

$$
\begin{aligned}
a_A^h(u,v) &= \int_{\Omega_F^h} \nabla u \cdot \nabla v \, dx\, dy - k^2 \int_{\Omega_F^h} uv \, dx\, dy \\
a_F^h(u,v) &= \int_{\Omega_A^h} \gamma_y \gamma_x^{-1} \partial_x u \, \partial_x v \, dA + \int_{\Omega_A^h} \gamma_x \gamma_y^{-1} \partial_y u \, \partial_y v \, dA - k^2 \int_{\Omega_A^h} \gamma_x \gamma_y uv \, dA \\
l^h(v) &= \int_{\Gamma^h} gv \, ds
\end{aligned}
\tag{B.33}
$$

In the above Γ^h is a piecewise linear approximation to curve Γ induced by T^h. We also define X_F^h and X_A^h to be the finite dimensional subsapces of X_F and X_A respectively.

The approximate solution to (B.31) is obtained by the Galerkin method outlined in appendix (A) which consists of finding $u_F^h \in X_F^h$ and $u_A^h \in X_A^h$ such that:

$$a_F^h(u_F^h, v^h) + a_A^h(u_A^h, v^h) = l^h(v^h) \tag{B.34}$$

where $u_F^h|_{\Gamma_I} = u_A^h|_{\Gamma_I}$ for all $v^h \in X^h$.

To obtain a finite set of equations relating the nodal values of u_F^h and u_A^h it is necessary to introduce the basis for X^h. This is the set of functions ϕ_i such that:

$$\phi_i(z_j) = \begin{cases} \delta_{ij} & \text{if} \quad z_i \quad \text{is a free vertex} \\ 0 & \text{if} \quad z_i \quad \text{is a constrained vertex} \end{cases}$$

and ϕ_i is non-zero only on triangle containing the vertex z_i. The index $i \in \{0, \ldots, N_v - 1\}$ where N_v is the total number of vertices in T^h corresponds to the unique vertex identification number. The solution u_F^h and u_A^h to (3.9) can be written as:

$$p_F = \sum_{i \in \Lambda_F} \alpha_i \, \phi_i \quad \text{and} \quad p_A = \sum_{i \in \Lambda_A} \alpha_i \, \phi_i \tag{B.35}$$

where Λ_F and Λ_A are the sets of vertex indices belonging to Ω_F^h and Ω_A^h respectively. Inserting (B.35) into (3.9) we obtain:

$$\sum_{i=0}^{N_v - 1} \alpha_i \left(a^h(\phi_i, \phi_j) + b^h(\phi_i, \phi_j) \right) = l^h(\phi_j) \tag{B.36}$$

for all $j = 0, \ldots, N_v - 1$. In matrix form, the above reads as:

$$\mathbf{K} \, \boldsymbol{\alpha} = \mathbf{f} \tag{B.37}$$

where $\mathbf{K}_{ij} = a^h(\phi_i, \phi_j) + b^h(\phi_i, \phi_j)$, $\boldsymbol{\alpha}_j = \alpha_j$ and $\mathbf{f}_j = l^h(\phi_j)$.

BIBLIOGRAPHY

[1] T.J. Alumbaugh and X. Jiao. Compact array-based mesh data structures. In B. Hanks, editor, *Proceedings of the 14th International Meshing Roundtable*, pages 485–503. Springer Berlin Heidelberg, 2005.

[2] S. Amini. On the choice of the coupling parameter in boundary integral formulations of the exterior acoustic problem. *Applied Analysis*, 35:75–92, 1989.

[3] S. Amini and P.J. Harris. On the Burton-Miller boundary integral formulation of the exterior acoustic problem. *Vibration and Acoustics*, 114:540–545, 1992.

[4] S. Amini and N.D. Maines. Preconditioned Krylov subspace methods for boundary element solution of the Helmholtz equation. *Numerical Methods in Engineering*, 41:875–898, 1998.

[5] X. Antoine, H. Barucq, and A. Bendali. Bayliss-turkel-like radiation condition on surfaces of arbitrary shape. *Journal of Mathematical Analysis and Applications*, 229:184–211, 1999.

[6] X. Antoine and M. Darbas. Integral Equations and Iterative Schemes for Acoustic Scattering Problems. Online: `http://hal.archives-ouvertes.fr/hal-00591456` - accessed July 25, 2014, May 2010.

[7] X. Antoine, M. Darbas, and Y.Y. Lu. An improved surface radiation condition for high-frequency acoustic scattering problems. *Computer Methods in Applied Mechanics and Engineering*, 195:4060–4074, 2006.

[8] W. Bangerth, R. Hartmann, and G. Kanschat. deal.II – a general purpose object oriented finite element library. *ACM Trans. Math. Softw.*, 33(4):24/1–24/27, 2007.

[9] W. Bangerth, T. Heister, L. Heltai, G. Kanschat, M. Kronbichler, M. Maier, B. Turcksin, and T. D. Young. The `deal.ii` library, version 8.1. *arXiv preprint*, 2013.

[10] P. Bastian, M. Blatt, A. Dedner, C. Engwer, R. Klfkorn, R. Kornhuber, M. Ohlberger, and S. Oliver. A generic grid interface for parallel and adaptive scientific computing. part ii: Implementation and tests in dune, 2007.

[11] P. Bastian, M. Blatt, A. Dedner, C. Engwer, M. Ohlberger, and O. Sander. A generic grid interface for parallel and adaptive scientific computing. part i: Abstract framework. *Computing*, 82(2):103–119, July 2008.

[12] A. Bayliss, M. Gunzburger, and E. Turkel. Boundary conditions for the numerical solution of elliptic equations in exterior regions. *SIAM Journal of Applied Mathematics*, 42:430–451, 1982.

[13] M. W. Beall and M. S. Shephard. A general topology-based mesh data structure. *International Journal for Numerical Methods in Engineering*, 40(9):1573–1596, 1997.

[14] E. Bécache, P.G. Petropoulos, and S.D. Gedney. On the long-time behavior of unsplit perfectly matched layers. *IEEE Transaction on Antennas and Propagation*, 52:1335–1342, 2004.

[15] A. Bendali and Y. Boubendir. Non-overlapping domain decomposition method for a nodal finite element method. *Numerical Mathematics*, 103:515–537, 2006.

[16] A. Bendali, Boubendir Y., and M. B. Fares. A FETI-like domain decomposition method for coupling finite elements and obundary elements in large-size problems of acoustic scattering. *Computer and Structures*, 85:526–535, 2007.

[17] J.P. Bérenger. A perfectly matched layer for the absorbtion of electromagnetic waves. *Journal of Computational Physics*, 114:185–200, 1994.

[18] A. Bermúdez, L. Hervella-Nieto, A. Prieto, and R. Rodriguez. An optimal perfectly matched layer with unbounded absorbing function for time-harmonic acoustic scattering problems. *Journal of Computational Physics*, 223:469–488, 2007.

[19] J. Billingham and A.C King. *Wave Motion*. Cambridge University Press, Cambridge, United Kingdom, 2000.

[20] Y. Botros and J. Volakis. Preconditioned generalized minimal residual iterative scheme for perfectly matched layer terminated applications. *IEEE Microwave and Guided Wave Letters*, 9(2):45–47, 1999.

[21] Y. Boubendir. An analysis of the BEM-FEM non-overlapping domain decomposition method for a scattering problem. *Journal of Computational and Applied Mathematics*, 204:282–291, 2007.

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

[23] 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. *International Journal of Mathematics and Mathematical Sciences*, 73:1624–1650, 2008.

[24] D. Braess. *Finite elements - Thery, fast solvers and applications in solid mechanics.* Cambridge University Press, Cambridge, 2002.

[25] S.C. Brenner and L.R Scott. *The Mathematical Theroy of Finite Element Methods.* Springer, New York, 2008.

[26] C. Burstedde, L. Wilcox, and O. Ghattas. `p4est`: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees. *SIAM Journal on Scientific Computing*, 33(3):1103–1133, 2011.

[27] K. Chen. On a class of preconditioning methods for dense linear systems from boundary elements. *SIAM Journal on Computational Science*, 20:684–698, 1998.

[28] K. Chen. An analysis of sparse approximate inverse preconditioners for boundary elements. *SIAM Journal on Matrix Analysis and Applications*, 22:1958–1978, 2001.

[29] E. Chow and Y. Saad. Approximate inverse preconditioner via sparse-sparse iterations. *SIAM Journal of Scientific Computing*, 19(3):995–1023, 1998.

[30] F. Collino and P.B. Monk. Optimizing the perfectly matched layer. *Computuer Methods in Applied Mechanics and Engineering*, 164:157–171, 1998.

[31] F. Collino and C. Tsogka. Application of the PML absorbing layer model to the linear elastodynamic problem in anisotropic heterogeneous media. *Geophysics*, 66(1):1294–1307, 2001.

[32] M. Costabel. Symmetric methods for the coupling of finite elements and boundary elements. In *Boundary Elements IV - Computational Mechanics*, volume 1, pages 411–420. Brebier, Southampton, 1987.

[33] B. Després. Domain decomposition method and the Helmholtz problem. In *Mathematical and Numerical Aspects of Wave Propagation Phenomena*. SIAM, 1991.

[34] Y. Dubois-Pe‘lerin and T. Zimmermann. Object-oriented finite element programming: Iii. an efficient implementation in c++. *Computer Methods in Applied Mechanics and Engineering*, 108(12):165 – 183, 1993.

[35] Y. Dubois-Plerin, T. Zimmermann, and P. Bomme. Object-oriented finite element programming: Ii. a prototype program in smalltalk. *Computer Methods in Applied Mechanics and Engineering*, 98(3):361 – 397, 1992.

[36] B. Engquist and A. Majda. Radiation boundary conditions for the numerical simulation of waves. *Mathematics of Computation*, 31:629–651, 1977.

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

[38] The MPI Forum. Mpi: A message passing interface, 1993. Online: `http://www.mcs.anl.gov/research/projects/mpi/` - accessed July 25, 2014.

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

[40] M.J Gander. Optimized schwarz methods for Helmholtz problems. In *Thirteenth international conference of domain decomposition*, pages 245–252, 2001.

[41] M.J Gander. Schwarz methods over the course of time. *Electronic Transactions on Numerical Analysis*, 31:228–255, 2008.

[42] C. Geuzaine and J.F. Remacle. Gmsh: A 3-d finite element mesh generator with built-in pre-and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331, 2009.

[43] D Givoli. *Numerical Methods for Problem in Infinite Domains.* Elsevier Science, 1992.

[44] M.S. Gockenbach. *Understanding and implementing the finite element method.* SIAM, Philadelphia, 2006.

[45] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *Journal of Computational Physics*, 73:325–348, 1987.

[46] J.S. Hestaven. On the analysis and construction of perfectly matched layers for the linearized euler equations. *Journal of Computational Physics*, 142:129–147, 1998.

[47] F.Q. Hu. A stable, perfectly matched layer for linearized euler equations in unsplit physical variables. *Journal of Computational Physics*, 173(2):455–480, 2001.

[48] F. Ihlenburg. *Finite Element Analysis of Acoustic Scattering.* Springer-Verlag, New York, NY, 1998.

[49] C. Johnson and J.C Nédélec. On the coupling of boundary integral and finite element methods. *Mathematics of Computation*, 35(152):1063–1079, 1980.

[50] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.*, 20(1):359–392, December 1998.

[51] J.B. Keller and D. Givoli. Exact non-reflecting boundary conditions. *Journal of Computational Physics*, 82:172–192, 1998.

[52] K. Kress and D. Colton. *Integral Equations Methods in Scattering Theory.* Wiley, New York, NY, 1983.

[53] K. Kress and D. Colton. *Inverse Acoustic and Electromagnetic Scattering Theroy.* Springer-Verlag, New York, NY, 1998.

[54] R. Kress. Minimizing the condition number of boundary integral operators in acoustic and electromagnetic scattering. *Journal of Mechanics and Applied Mathematics*, 38:323–341, 1985.

[55] R. Kress and W.T. Spassov. On the condition number of boundary integral operators in acoustic and electromagnetic scattering. *Numerical Mathematics*, 42:77–95, 1983.

[56] Xiaoye S. Li. An overview of SuperLU: Algorithms, implementation, and user interface. *ACM Transactions on Mathematical Software*, 31(3):302–325, September 2005.

[57] P.L. Lions. On the schwartz alternating method I. In *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 1–42. SIAM, 1988.

[58] P.L. Lions. On the schwartz alternating method III - a variant for nonoverlapping subdomains. In *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 202–223. SIAM, 1990.

[59] R. MacCamy and S. Marin. A finite element method for exterior interface problems. *International Journal of Mathematics and Mathematical Sciences*, 3:311–350, 1980.

[60] B.H. McDonald and A. Wexler. Finite element solution of unbounded field problems. *IEEE Transaction on Microwave Theory and Techniques*, MTT-20:841–847, 1972.

[61] R. Mittra and O. Ramahi. Absorbing boundary conditions for the direct solution of partial differential equations arising in electromagnetic scattering problems. In *Progress in electomagnetic Research*, pages 133–173. Elsevier, Newark, NJ, 1990.

[62] F. Nataf, F. Rogier, and E. DeSturler. Optimal interface conditions for domain decomposition methods. Technical Report 301, Ecole Polytechnique, 1994. CMAP.

[63] I.M. Navon, B. Neta, and M.Y. Hussaini. A perfectly matched layer formulation for the nonlinear shallow water equation models: the split equation approach. *Montly Weather Review*, 2001.

[64] P.G. Petropoulos. On the termination of the perfectly matched layer with local absorbing boundary conditions. *Journal of Computational Physics*, 143:665–673, 1998.

[65] P.G. Petropoulos. Reflectionless sponge layers as absorbing boundary conditions for the numerical solution of maxwell equations in rectangular, cylindrical, and spherical coordinates. *SIAM Journal of Applied Mathematics*, 60(3):1037–1058, 2000.

[66] P.J. Plauger, M. Lee, D. Musser, and A.A. Stepanov. *C++ Standard Template Library*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 2000.

[67] A. Quarteroni and A. Valli. *Domain Decomposition Methods for Partial Differential Equations*. Clarendon Press, Oxford, 1999.

[68] V. Rokhlin. Rapid solution of integral equations of scattering theory in two dimensions. *Journal of Computational Physics*, 6(2):414–439, 1990.

[69] F. Roux, F. Magoulés, S. Salmon, and L. Series. Optimization of interface operators based on algebraic approach. In *Domain Decomposition Methods in Science and Engineering*, pages 297–304. 2002.

[70] F. Roux, L. Magoulés, L. Series, and Y. Boubendir. Approximation of optimal interface boundary conditions for two-lagrange multiplier feti method. In *Lecture Notes in Computational Science and Engineering*, volume 40, pages 283–290. 2005.

[71] Y. Saad. *Iterative Methods for sparse linear systems*. SIAM, 2nd edition, 2003.

[72] Y. Saad and M. H. Schultz. Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7(3):857–868, 1986.

[73] M.E. Taylor. *Pseudodiffetential Operators*. Princeton University Press, Princeton, 1981.

[74] A. Toselli and O. Widlund. *Domain Decomposition Methods - Algorithms and Theory*. Springer, Berlin, Germany, 2005.

[75] L.N. Trefethen and D. Bau. *Numerical Linear Algebra*. SIAM, 1997.

[76] S.V. Tsynkov. Numerical solution of problems on unbounded domains. A review. *Applied Numerical Methods*, 27:465–532, 1998.

[77] T. Zimmermann, Y. Dubois-Plerin, and P. Bomme. Object-oriented finite element programming: I. governing principles. *Computer Methods in Applied Mechanics and Engineering*, 98(2):291 – 303, 1992.