ABSTRACT

DOMAIN DECOMPOSITION METHODS FOR LINEAR AND NON-LINEAR ELLIPTIC PROBLEMS

by
Tadanaga Takahashi

This dissertation develops domain decomposition methods (DDMs), which reformulate a boundary value problem as a collection of local problems on subdomains. The first project concerns the Helmholtz equation in wave scattering. The proposed method adapts a non-overlapping finite element DDM with continuous cross-points. The unbounded computational domain is truncated, and the near-field wave is computed using a parallel finite element method. Several improvements to the standard transmission operator are developed. The resulting methods accelerate convergence and work with both perfectly matched layers and absorbing boundary conditions.

The second project explores DDMs for the Monge–Ampère (MA) equation, a fully nonlinear, second-order, degenerate elliptic equation. Recent advances have produced new discretization schemes for this equation, but solvers for the resulting nonlinear systems have received less attention. Several DDMs are combined with convergent monotone schemes for viscosity solutions. Numerical experiments in one and two dimensions investigate convergence as the degree of degeneracy, number of subdomains, overlap percentage, and discretization parameters vary.

DOMAIN DECOMPOSITION METHODS FOR LINEAR AND NON-LINEAR ELLIPTIC PROBLEMS

by
Tadanaga Takahashi

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
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
Department of Mathematics and Computer Science, Rutgers-Newark

May 2023

Copyright © 2023 by Tadanaga Takahashi
ALL RIGHTS RESERVED

NOTE ON THIS REVISED EDITION

This website edition was revised on November 10, 2026, from the dissertation submitted in May 2023. The revisions correct typographical errors and improve the wording and presentation. Personal and administrative front matter has been omitted. The research reflects the original dissertation; this edition does not incorporate subsequent research or new numerical results. The official submitted dissertation remains the version of record.

TABLE OF CONTENTS

Chapter **Page**

1 INTRODUCTION 1

1.1 Boundary Value Problem 1

1.2 Domain Decomposition Method 2

1.3 Helmholtz Equation 4

1.4 Monge-Ampère Equation 5

2 DOMAIN DECOMPOSITION FOR THE HELMHOLTZ EQUATION . . 8

2.1 Models 8

2.1.1 Electromagnetic Waves 8

2.1.2 Scattering Problem 9

2.1.3 Nonhomogeneous Scattering Model 11

2.1.4 Truncated Model 12

2.2 Domain Decomposition Formulations 14

2.2.1 Direct Method for the Nonhomogeneous Problem 14

2.2.2 DDM for the Heterogeneous Problem 15

2.2.3 Non-overlapping DDM with Robin Conditions 17

2.2.4 Domain Decomposition as a Preconditioner 20

2.3 Transmission Operators 21

2.3.1 Examples of Transmission Operators 22

2.3.2 Dual Transmission Operator 23

2.4 Domain Decomposition at the Discrete Level 25

2.4.1 Weak Form 26

2.4.2 The Mesh 28

2.4.3 Continuous Cross-point Algorithm 29

3 CONTRIBUTIONS TO HELMHOLTZ DDM 33

3.1 Padé-localized Square Root Operator 33

TABLE OF CONTENTS
(Continued)

Chapter **Page**

3.1.1 Padé Approximation of the Square Root Operator 33

3.1.2 Padé Operator with Cross-points 35

3.1.3 Detailed Example 38

3.1.4 Numerical Results for the Padé Operator 43

3.2 Boundary Integral Equation Transmission Operator 45

3.2.1 DDM with IE Transmission Operator 46

3.2.2 Concentric Example 48

3.2.3 Adaptation of the IE Operator with Cross-points 50

3.2.4 Results for General Interfaces 51

3.3 Perfectly Matched Layer 53

3.3.1 Convergence on the Half-plane 54

3.3.2 The Cartesian PML Problem 62

3.3.3 Numerical Tests for the PML 63

3.4 The Dual Transmission Operator 64

3.4.1 Motivation for the Dual Transmission Operator 65

3.4.2 Dual Transmission Padé Operator 70

3.4.3 Numerical Results for the Dual Transmission Operators 71

3.5 Adaptive Radiation Condition 73

3.5.1 Numerical Results 78

3.6 Acceleration of the Convergence of the NDDM Algorithm 79

3.6.1 Numerical Results 80

3.6.2 Comparison with other Transmission Operators 81

3.6.3 Partially Coated Scatterer 83

3.7 Concluding Remarks 85

4 BACKGROUND FOR THE MONGE-AMPÈRE EQUATION 87

4.1 Viscosity Solution 87

TABLE OF CONTENTS
(Continued)

Chapter **Page**

4.1.1 Dirichlet Problem 89

4.2 Discretization 90

4.2.1 Monotone Discretization 91

4.2.2 Wide Stencil Methods 92

4.2.3 Determinant Integral Representation Quadrature 94

4.3 Nonlinear Methods and Domain Decomposition 97

4.3.1 Newton's Method 98

4.3.2 Nonlinear Preconditioning 99

4.3.3 Nonlinear Additive Schwarz 101

4.3.4 Nonlinear Preconditioning with Domain Decomposition 102

4.3.5 Newton-Krylov-Schwarz 103

4.3.6 Convergence of the NASM for the Monge-Ampère Equation . . 106

5 DOMAIN DECOMPOSITION METHODS FOR THE MONGE-AMPÈRE EQUATION 109

5.1 Domain Decomposition for the One-dimensional Monge-Ampère Equation 109

5.1.1 One-dimensional Model Problem 110

5.1.2 Non-overlapping Domain Decomposition 112

5.1.3 Overlapping Domain Decomposition 115

5.1.4 Discretization 116

5.1.5 Implementation Details 117

5.1.6 Results for the Non-overlapping Method 118

5.1.7 Results for the Overlapping Method 120

5.1.8 Concluding Remarks 122

5.2 Domain Decomposition Methods for the Two-dimensional Monge-Ampère Equation 123

5.2.1 Two-dimensional Model Problem 123

5.2.2 Problem Parameters 124

TABLE OF CONTENTS
(Continued)

Chapter **Page**

5.2.3 Serial Algorithm 126

5.2.4 NKS and SNK 127

5.2.5 Numerical Results 128

5.2.6 Comments on the Results 141

5.3 Concluding Remarks 142

APPENDIX A HELMHOLTZ 144

A.1 Fundamental Exterior Problem 144

A.1.1 Plane Wave 145

A.1.2 Exterior Solution 147

A.2 Scattering Problem on the Annulus 147

A.3 Generalized Problem and the Bilinear Form 150

A.4 Laplace-Beltrami on the Interface 151

A.4.1 Laplace-Beltrami with Domain Decomposition 152

A.5 Weak NDDM-FEM Algorithm 155

A.6 Domain Decomposition with the Square Root Operator 157

APPENDIX B MONGE-AMPÈRE 161

B.1 PETSc's Cartesian Mesh 161

B.2 Computing the Second Directional Derivative 163

REFERENCES 166

LIST OF TABLES

Table		Page
3.1	Iteration Number with respect to n_λ	44
3.2	Iteration Number with respect to k	44
3.3	Iteration Number with respect to N_d	44
3.4	Iteration Numbers for the Layered Configuration	45
3.5	Iteration Numbers for n_λ and k Tests on the Concentric Domain	49
3.6	Iteration Numbers for n_λ and k Tests for the Pie Domain	52
3.7	N_d Test for the Pie Domain	52
3.8	The n_λ and k Tests for the Layered Configuration	52
3.9	Iteration Number with respect to Different Operators and k	64
3.10	Iteration Numbers for the Scalar Dual Operators	69
3.11	Comparison of Iteration Numbers with and without the Dual Transmission Operator	72
3.12	Iteration Numbers for the ARC with $k = \pi$	79
3.13	Iteration numbers with respect to δ	82
3.14	Iteration Numbers for Fixed δ	82
5.1	Performance of the NDDM for Various β	119
5.2	Performance of the Non-optimized NDDM	120
5.3	Performance of the NDDM with Optimized β	121
5.4	Comparison of the Two Residual Definitions	121
5.5	Performance of the 1D ODDM	122
5.6	Stats for the 2D Serial Algorithm with $N = 200$	129
5.7	Final Error of the NK Solver with respect to L and Tolerance	134
5.8	Results for NK on Example 1 with Tolerance 10^{-8}	134
5.9	Results for NKS on Example 1 on 4 Subdomains	134
5.10	Results for SIN on Example 1 on Four subdomains	135

LIST OF TABLES
(Continued)

Table **Page**

5.11 Results for HTN on Example 1 on 4 Subdomains 135

5.12 Runtime of the NK method for example 5 with respect to mesh size. . . 138

5.13 Results for the Serial Algorithm on Example 5 139

5.14 DDM Performance on the Large Scale Problem 140

5.15 Resolution $h = 0.05$, Overlap $p = 40\%$, Newton Tolerance 10^{-2}, and Krylov Tolerance 10^{-1} 140

5.16 Resolution $h = 0.05$, Overlap $p = 10\%$, Newton Tolerance 10^{-3}, and Krylov Tolerance 10^{-1} 141

LIST OF FIGURES

Figure		Page
2.1	Sketch of a basic scattering problem.	10
2.2	Sample geometry of the model problem. Ω represents the dielectric. Γ is a wave-reflecting core. Σ is a wave-penetrable surface.	11
2.3	N-Subdomain DDM. The cross-points are shown in red.	18
2.4	A multilayered scatterer with constant index of refraction in each layer.	24
2.5	Example of a NDDM mesh with triangular elements.	28
3.1	Supplementary figures to demonstrate the implementation of the PLS operator assembly.	39
3.2	Shows the edges e_6 marked in blue and e_{12} marked in blue.	41
3.3	Geometries used for the numerical simulation: pie (left) and layered (right) configurations.	43
3.4	Sample geometry of a DDM with scatterer Γ and truncating surface Σ.	46
3.5	Two-subdomain geometry with $S = \Sigma_{ij}$ as the closed curve.	48
3.6	These plots show the iteration number of the complexified IE operator with respect to $\epsilon \in (0, 1)$ for the two-subdomain concentric geometry.	50
3.7	Sketch of the domain for the original problem Equation Equation (3.33) and the truncated model Equation Equation (3.34).	54
3.8	Domain of the PML problem taken from [53].	62
3.9	Spectrum of the Després and the EMDA operators with $n_1 = 0.7$ and $n_2 = 1.3$ are plotted. The maximum $M = \max_{\xi>0} \rho(\xi)$ is plotted as a filled circle.	67
3.10	Plot shows the spectrum for the dual operators and their peak values.	68
3.11	Geometry to test the dual transmission EMDA operators.	69
3.12	Numerical solutions for the scattering problem with and without homogeneity.	69
3.13	Two-layer nonhomogeneous scattering geometry. The material layers and the dashed artificial interfaces divide the computational domain into eight subdomains and introduce cross-points.	72

LIST OF FIGURES
(Continued)

Figure **Page**

3.14 Sketch of a scattering geometry with Γ being the scatterer and Σ being the truncating surface. 73

3.15 Sketch of the NDDM-ARC domain. The collection of artificial interfaces in bold dashed lines can be used for the ARC. 75

3.16 Sketch of the NDDM-ARC domain for the numerical tests. 78

3.17 A comparison of the error between the ABC and the ARC. 79

3.18 Quantity of nodes on each boundary are shown in all test cases 81

3.19 Plots indicate the iteration number for the four test cases with respect to δ and for the three transmission operators. 83

3.20 Geometry for partially coated scatterer. 84

3.21 The discretization of the interfaces are shown on the left and the real part of the numerical solution on the right. 85

3.22 Iteration numbers for the NDDMs on the partially coated scatterer with respect to δ and for the different transmission operators. 85

4.1 Example of a wide stencil with width levels of 1, 2, and 3. 94

4.2 The L^1 stencil with $d = 3$. This figure was taken from [22]. 96

5.1 The osculating circle of $f(x) = x^3$ at $x = 0.5$. 110

5.2 Three cases for the Gaussian curvature problem. 111

5.3 Pointwise error and runtime for the three cases. 112

5.4 Diagram of non-overlapping domain decomposition. 113

5.5 Overlapping subdomains. 115

5.6 Iteration numbers are plotted with respect to β for two subdomains (left) and four subdomains (right). 120

5.7 Overlapping decomposition with $N_d = 4$ subdomains 125

5.8 Runtime comparison for the four examples on a $N \times N$ mesh. 127

5.9 Runtime (left) and iteration number (right) for Example 3 with respect to relaxed Krylov and Newton tolerances. 130

5.10 The plots show how under-solving affects the performance of the NKS. . 130

5.11 Comparison of the HTN and SIN with respect to subdomain overlap. . . 131

LIST OF FIGURES
(Continued)

Figure **Page**

5.12 Performance of the NKS with respect to overlap. 132

5.13 Strong scaling test with $N = 300$. 137

5.14 Weak scaling test. 137

5.15 Iteration and runtime for example 5 with respect to subdomain overlap. 139

A.1 Exterior Scattering problem. 144

A.2 Exterior scattering problem truncated by an artificial surface Σ. 148

A.3 A domain with three subdomains. 153

CHAPTER 1

INTRODUCTION

This dissertation develops efficient numerical methods for boundary value problems. In numerical analysis, algorithms are used to approximate solutions of partial differential equations (PDEs). The methods introduced here use domain decomposition to improve computational efficiency. This chapter gives an overview of domain decomposition methods and the problems to which they are applied.

1.1 Boundary Value Problem

This dissertation develops domain decomposition methods for two time-independent problems. The first involves the Helmholtz equation, a non-coercive linear elliptic equation. The second involves the Monge-Ampère equation, a fully nonlinear elliptic equation. Both are formulated as boundary value problems (BVPs).

Consider an unknown function u defined on a domain Ω. A known, possibly nonlinear partial differential operator F governs its behavior in the interior. To determine u, we also prescribe boundary conditions on $\partial\Omega$. We denote the source term by f, the boundary operator by B, and the boundary data by g. The goal is to find u satisfying both the differential equation and the boundary conditions. We write the BVP as

$$\mathscr{B}(u; f, g) = \begin{cases} F(u) = f, & \text{in } \Omega \\ B(u) = g, & \text{on } \partial\Omega \end{cases} \tag{1.1}$$

Equation (1.1) asks for u given the data f and g. In the context of DDM, we call it the original problem. The operator F is often motivated by physics, and computing u provides a **simulation** of the modeled phenomenon. We consider well-posed BVPs:

a solution exists, is unique, and depends continuously on the data f and g. Our main concern is how to approximate that solution efficiently.

1.2 Domain Decomposition Method

Domain decomposition methods (DDMs) form a broad class of algorithms that solve BVPs by partitioning the domain Ω. Suppose Ω is split into N_d subdomains Ω_i, $i = 1, \ldots, N_d$. These subdomains may overlap or may be disjoint, i.e., $\Omega_i \cap \Omega_j = \emptyset, \forall i \neq j$. Instead of solving the original problem over Ω, we seek the same solution by solving smaller local BVPs $\mathscr{B}_i(u_i; f_i, g_i)$ over Ω_i.

Classical DDMs are iterative algorithms in which the local problems $\mathscr{B}_i$ are solved repeatedly with updated boundary data $g_i^{(n)}$, where n is the iteration index. For a convergent method, the local iterates approach the restrictions of the original solution, i.e., $u_i^{(n)} \to u_i, \forall i$. A schematic iteration is outlined in Algorithm 1. The local problems are designed to be easier to solve than the original problem, allowing the global solution to be obtained through a sequence of smaller solves.

Algorithm 1: Classical DDM

Let $n = 1$. Let $u_i^{(0)} = 0$, $\forall i$;
while *not converged* **do**
  Update boundary data $g_i^{(n)}$ based on previous solutions $u_i^{(n-1)}$ $\forall i$;
  Solve local problem $\mathscr{B}_i(u_i^{(n)}; f_i, g_i^{(n)})$, $\forall i$;
  n++;
end

DDM is also an important research area in high-performance computing (HPC) [63, 59, 34]. Multicore computers can execute independent tasks simultaneously. Domain decomposition distributes the algebraic systems associated with subdomains among processing units, allowing local problems to be solved in parallel. As computer architectures have shifted toward greater parallelism, DDMs have become useful tools for exploiting machines with many processing units.

Parallel algorithms support both faster computation and distributed memory. Strong scaling measures the reduction in runtime for a fixed problem as computational resources increase. Weak scaling measures performance as the problem size grows with the available resources. Because each node has its own memory, distributing a problem also increases the total memory available. For example, if each node provides 16 GB, a problem distributed over N_d nodes can use memory on the order of $N_d \times 16$ GB. Some simulations are too large to run without distributed memory. One example is the cosmological simulation performed on the Titan supercomputer at Oak Ridge National Laboratory using 16,384 nodes, each with 32 GB of RAM [46].

Although modern hardware strongly motivates DDMs, domain decomposition originated as an analytical tool. In 1870, Hermann Schwarz introduced the alternating procedure to establish existence results for the Laplace equation on composite domains. Gander and Gerhard recount its origins in [43]. Later work developed Schwarz methods as practical numerical algorithms [51]. Decomposing a problem into local components fits naturally with parallel computing, and many numerical techniques have since emerged from this framework. Early work focused on linear elliptic BVPs, such as the Laplace equation; subsequent research has extended DDMs to a much broader range of problems.

This dissertation contributes to DDMs for the Helmholtz and Monge-Ampère equations. Domain decomposition for the Helmholtz equation has been studied since the early 1990s by Després and others [32]. Developments include wavenumber-dependent transmission operators [15, 42] and approximations of the radiation condition [8]. Domain decomposition for the Monge-Ampère equation is less developed. Its fully nonlinear operator $F(u)$ lies outside much of the existing DDM theory, but the problem can still benefit from domain decomposition solvers. The contributions presented here include new Helmholtz DDM techniques and an investigation of DDMs for the Monge-Ampère equation.

1.3 Helmholtz Equation

This section introduces the Helmholtz equation, its applications, and the motivation for domain decomposition. The equation is

$$\Delta u + k^2 u = 0 \tag{1.2}$$

where $u \in \mathbb{C}^2$ is the complex wave amplitude and $k \in \mathbb{R}$ is the wavenumber. The two-dimensional (2D) Laplace operator is defined as $\Delta = \frac{\partial}{\partial x} + \frac{\partial}{\partial y}$. The Helmholtz equation describes the time-independent amplitude of a time-harmonic wave. It models mechanical waves, including sound and seismic waves, as well as electromagnetic waves. With appropriate boundary conditions, it describes reflection, refraction, and absorption by an object. The corresponding BVP is called a direct scattering problem. Wave scattering simulations support applications in telecommunications, radar, and noninvasive imaging. An object in a wave field changes the observed wave pattern. Conversely, the resulting pattern can be used to infer properties of the object; this is the subject of inverse scattering.

High-frequency scattering simulations are computationally demanding. As the frequency increases, more oscillations fit within a fixed region. Resolving these oscillations requires a finer discretization and therefore a larger algebraic system.

Domain decomposition is an important approach to efficient scattering simulation. The volume *Modern Solvers for Helmholtz Problems* [49] discusses DDMs alongside multigrid and boundary element methods. A principal advantage is the ability to reduce runtime through parallel computation. Transmission operators have been developed whose convergence behavior is robust with respect to wave frequency [17]. Domain decomposition also provides a framework for coupling different local solvers, including finite element and boundary element methods. This flexibility is useful for problems such as scattering in multiple media [17].

Chapter 2 reviews the scattering problem and relevant domain decomposition techniques, culminating in the continuous cross-point algorithm for non-overlapping DDM. Chapter 3 presents the Helmholtz contributions built on this framework.

1.4 Monge-Ampère Equation

This section introduces the Monge-Ampère equation, its applications, and the motivation for domain decomposition. The equation

$$\det\left(D^2u(x)\right) = f(x), \quad \text{for } x \text{ in } \Omega \tag{1.3}$$

is a nonlinear second-order partial differential equation (PDE). The operator D^2 denotes the Hessian of u. In two dimensions, $\det(D^2u) = u_{xx}u_{yy} - u_{xy}u_{yx}$. For $x \in R^n$ with $n > 1$, the equation is fully nonlinear. Convexity is imposed on u to select the elliptic branch of the equation.

The Monge-Ampère equation has numerous applications. One example is the Gaussian curvature problem on $\mathbb{R}^d$:

$$\begin{cases} \det\left(D^2u(x)\right) = \kappa(x)\left(1 + |\nabla u(x)|^2\right)^{(d+2)/2}, & x \in \Omega \\ u(x) = g(x), & x \in \partial\Omega \\ u \text{ is convex.} \end{cases} \tag{1.4}$$

The unknown surface u is prescribed through its Gaussian curvature and boundary data. This problem is related to developable surfaces, which are important in manufacturing objects from flat materials such as fabric, paper, or sheet metal [50]. Other applications of the Monge-Ampère equation include optical lens design [38], mesh generation [23], and machine learning [58].

The Monge-Ampère equation is closely linked to continuous optimal transport. This problem seeks a map $T(x)$ that transports the measure μ on X to ν on Y while

minimizing

$$\int_X c(x, T(x)) d\mu(x), \tag{1.5}$$

where $c : X \times Y \to [0, \infty)$ is the cost. Among all admissible transport maps, the optimal map minimizes the prescribed transport cost. For example, supplies may need to be moved from warehouses to stores, with a cost determined by travel distance or fuel consumption.

When the cost function is quadratic,

$$c = |x - T(x)|^2$$

, the transport problem is related to a Monge-Ampère BVP. Under the appropriate transport formulation, the solution to Equation (1.3) is a potential for the optimal map in Equation (1.5), i.e., $\nabla u = T$. Solving the associated Monge-Ampère equation can be simpler than solving the transport problem directly. Villani discusses the history, mathematical framework, and applications of optimal transport in [64].

The nonlinearity of the Monge-Ampère equation creates significant numerical challenges. Viscosity solutions provide a weak solution framework when classical solutions do not exist [29]. Standard discretizations may fail to converge to singular viscosity solutions. For degenerate elliptic PDEs, monotonicity is a key condition in convergence theory [7]. Wide stencil schemes address the need for monotone discretizations [56], but their accuracy can be lower than that of standard schemes. Depending on the method and assumptions, accuracy may be $O(h)$, $O(\sqrt{h})$, or $O(h^{4/3})$ [28, 40, 22]. Angular resolution introduces an additional error component; an example is discussed in [28]. Fine meshes may therefore be needed to obtain adequate accuracy, increasing the computational cost. Further discussions of discretizations for the Monge-Ampère equation can be found in [9, 39].

Efficient solvers are therefore valuable for the Monge-Ampère equation. Explicit iterations based on forward Euler can be proved convergent, but their restrictive time steps can make them slow, particularly for singular solutions and in three dimensions (3D) [57]. Wide stencil discretizations can also be solved using Newton's method. Convergence requires a sufficiently accurate initial guess, which can be difficult to obtain for nonsmooth problems. In three dimensions, solving the Jacobian systems can also be prohibitively expensive [41].

Domain decomposition offers several approaches to reducing this cost. An additive Schwarz method can be applied directly to the nonlinear BVP, using a monotone discretization on each subdomain. Alternatively, domain decomposition can precondition the linear systems arising in Newton's method. Both approaches can reduce runtime relative to a serial solver. They also raise theoretical questions: convergence theory is established for some quasilinear and semilinear problems, but is less developed for fully nonlinear equations. DDMs also allow different local solvers to be used in regions with different PDE properties [17]. For the Monge-Ampère equation, this flexibility may be useful near regions of degeneracy or boundary singularities.

Chapter 4 reviews monotone discretizations for the Monge-Ampère equation. Chapter 5 presents the domain decomposition contributions and numerical experiments.

CHAPTER 2

DOMAIN DECOMPOSITION FOR THE HELMHOLTZ EQUATION

This chapter introduces scattering models and domain decomposition techniques for the Helmholtz BVP. The discussion leads to the non-overlapping DDM with continuous cross-points, which provides the foundation for the contributions in Chapter 3.

2.1 Models

This section introduces electromagnetic wave scattering and its formulation as a boundary value problem. The Helmholtz equation describes the time-independent amplitude of a time-harmonic wave.

2.1.1 Electromagnetic Waves

Electromagnetic (EM) waves propagating through a vacuum consist of magnetic field $\mathbf{M}$ and an electric field $\mathbf{E}$ oscillating in phase with each other. Both fields are modeled by the wave equation:

$$\begin{aligned} \left(\tfrac{1}{\mu_0\varepsilon_0}\nabla^2 - \tfrac{\partial^2}{\partial t^2}\right)\mathbf{E} &= \mathbf{0} \\ \left(\tfrac{1}{\mu_0\varepsilon_0}\nabla^2 - \tfrac{\partial^2}{\partial t^2}\right)\mathbf{B} &= \mathbf{0} \end{aligned} \tag{2.1}$$

where μ_0 and ε_0 are the permeability and permittivity of free space, and ∇^2 is the Laplace operator. These equations follow from Maxwell's equations [47]. The wave speed is $c = 1/\sqrt{\mu_0\varepsilon_0}$. In a medium with permeability μ and permittivity ε, the speed is $v = 1/\sqrt{\mu\varepsilon}$ and the refractive index is $n = c/v$. Each component of the electric field satisfies the wave equation.

Consider just the z-component of the wave equation for the electric field. By introducing the ansatz

$$E(\mathbf{x}, t) = \Re\left\{u(\mathbf{x})e^{-i\omega t}\right\}, \tag{2.2}$$

with frequency $\omega > 0$ into $\left(c^2\nabla^2 - \frac{\partial^2}{\partial t^2}\right) E = 0$ gives a PDE for the complex amplitude $u(\mathbf{x})$:

$$\Delta u + k^2 u = 0 \tag{2.3}$$

where $k = \omega/c$ is the wavenumber. Thus, the Helmholtz equation governs the spatial amplitude of a time-harmonic wave.

2.1.2 Scattering Problem

In a direct scattering problem, an incident wave u^{inc} illuminates an object, and the resulting scattered wave u^{sc} is sought. The scattered wave propagates outward from the object. The propagation domain is denoted by Ω, and the total wave u is the sum of the incident and scattered waves:

$$u = u^{\text{inc}} + u^{\text{sc}}. \tag{2.4}$$

The incident wave is a problem parameter which is free to choose. The incident wave is usually chosen to be the plane wave

$$u^{\text{inc}} = e^{i\mathbf{k}\cdot\mathbf{x}} \tag{2.5}$$

where $\mathbf{k}$ is a wave vector in the direction of propagation satisfying $|\mathbf{k}| = k$. Both this plane wave and $u^{\text{inc}} = e^{-i\mathbf{k}\cdot\mathbf{x}}$ satisfy the Helmholtz equation. To select the outgoing scattered solution, a condition at infinity is required. The classical choice is the Sommerfeld radiation condition:

$$\lim_{|x|\to+\infty} |x|^{\frac{d-1}{2}} \left(\nabla\left(u_0 - u^{\text{inc}}\right) \cdot \frac{x}{|x|} - ik\left(u_0 - u^{\text{inc}}\right)\right) = 0, \tag{2.6}$$

where d is the dimension. This condition requires the wave to be outgoing. There are other variations but the Sommerfeld condition is the most fundamental.

A boundary condition is also needed on the surface of the scatterer $\Gamma = \partial\Omega$. Borrowing terminology from acoustic scattering, sound-soft boundary condition refers to zero Dirichlet condition. This is expressed as $u = 0$, or equivalently $u^{\text{sc}} = -u^{\text{inc}}$. This condition states that upon impact, a wave of opposite polarity and equal magnitude is created. The sound-hard boundary condition refers to the zero Neumann condition. This is expressed as $\partial_{\mathbf{n}} u = 0$ or $\partial_{\mathbf{n}} u^{\text{sc}} = -\partial_{\mathbf{n}} u^{\text{inc}}$. This condition describes the total reflection of the surface. Last is the Robin boundary condition, also known as the impedance condition. It can be used to model imperfection in the material's conductive properties [12].

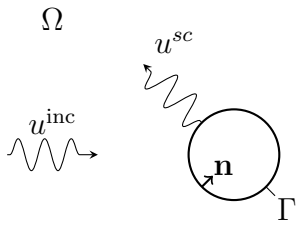


Figure 2.1 Sketch of a basic scattering problem.

The scattering BVP can now be written. The total wave u satisfies

$$\begin{cases} \Delta u + k^2 u = 0 & x \in \Omega \\ \beta_1 u + \partial_{\mathbf{n}} u = 0 & x \in \Gamma \\ |x|^{1/2} \left(\nabla u \cdot \dfrac{x}{|x|} - iku \right) = 0 & |x| \to +\infty. \end{cases} \tag{2.7}$$

The condition on Γ is a Robin condition with constant coefficient β_1. Figure 2.1 shows a sketch of the geometry of the problem in 2D. The wave is propagating in free space. It is sometimes beneficial to define the region interior of the scatterer $S = \mathbb{R}^2 - \Omega$. The wave is undefined in S.

2.1.3 Nonhomogeneous Scattering Model

Now consider a scatterer surrounded by a penetrable coating, such as a dielectric. Waves refract and reflect as they cross the coating boundary. We seek u and u_0, the z-components of the total magnetic field inside and outside the dielectric, respectively. Figure 2.2 shows an example geometry. The free-space region is Ω_0, and the possibly nonuniform dielectric occupies Ω. The scatterer, with boundary Γ, is embedded in the dielectric, whose outer surface is Σ.

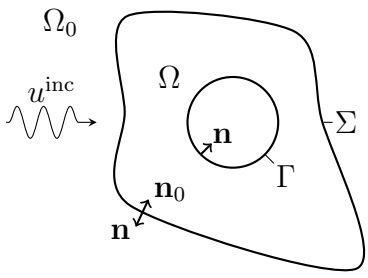


Figure 2.2 Sample geometry of the model problem. Ω represents the dielectric. Γ is a wave-reflecting core. Σ is a wave-penetrable surface.

Written in terms of the relative permittivity ε and the index of refraction n, the system is expressed as follows:

$$\begin{aligned} \nabla \cdot \left(\frac{1}{\varepsilon}\nabla u\right) + k^2\frac{n^2}{\varepsilon}u = 0 \text{ in } \Omega \subset \mathbb{R}^2 \\ \Delta u_0 + k^2 u_0 = 0 \text{ in } \Omega_0 \subset \mathbb{R}^2 \\ \varepsilon^{-1}\partial_{\mathbf{n}} u = 0 \text{ on } \Gamma \\ |x|^{1/2}\left(\nabla\left(u_0 - u^{\text{inc}}\right)\cdot\frac{x}{|x|} - \mathrm{i}k\left(u_0 - u^{\text{inc}}\right)\right) = 0 \text{ as } |x| \to +\infty \end{aligned} \tag{2.8}$$

The interior region Ω is modeled by the variable coefficient Helmholtz and the exterior region Ω_0 by the regular Helmholtz. The two equations are coupled at Σ; along this interface we have the transmission conditions:

$$\begin{aligned} u_0 - u = 0 \text{ on } \Sigma \\ \partial_{\mathbf{n}_0} u_0 + \varepsilon^{-1}\partial_{\mathbf{n}} u = 0 \text{ on } \Sigma \end{aligned} \tag{2.9}$$

Equations (2.8) and (2.9) specify the scattering problem for the total wave. Since the incident wave is known, we rewrite the system in terms of the scattered fields u^{sc} and u_0^{sc}:

$$\begin{aligned}
\nabla \cdot \left(\frac{1}{\varepsilon}\nabla u^{\text{sc}}\right) + k^2 \frac{n^2}{\varepsilon} u^{\text{sc}} &= f_\Omega && \text{in } \Omega \\
\Delta u_0^{\text{sc}} + k^2 u_0^{\text{sc}} &= 0 && \text{in } \Omega_0 \\
\varepsilon^{-1} \partial_{\text{n}} u^{\text{sc}} &= f_\Gamma && \text{on } \Gamma \\
u^{\text{sc}} - u_0^{\text{sc}} &= 0 && \text{on } \Sigma \\
\varepsilon^{-1} \partial_{\text{n}} u^{\text{sc}} + \partial_{\mathbf{n}_0} u_0^{\text{sc}} &= 0 && \text{on } \Sigma \\
|x|^{1/2} \left(\nabla u_0^{\text{sc}} \cdot \frac{x}{|x|} - iku_0^{\text{sc}}\right) &= 0 && \text{as } |x| \to +\infty,
\end{aligned} \tag{2.10}$$

where f_Ω and f_Γ are known terms determined by u^{inc}. In particular, $f_\Omega = 0$ when $n = \varepsilon = 1$. To simplify notation, we continue to use the total field where convenient. Any incident field satisfying the Helmholtz equation may be used; the usual choice is $u^{\text{inc}} = e^{i\mathbf{k}\cdot\mathbf{x}}$. An exact solution is available for an annulus centered at the origin when $\varepsilon = 1$ and $n = 1$. The wavenumber and wavelength are related by $k = 2\pi/\lambda$. Under mild assumptions on n, ε, Γ and Σ, Equation (2.10) is well-posed [65].

2.1.4 Truncated Model

The direct scattering problem is posed on an unbounded domain, which cannot be discretized by conventional finite element techniques. A natural approach is to truncate the domain and impose an approximate radiation condition on an artificial boundary a finite distance from the scatterer. This introduces truncation error but makes the problem accessible to methods defined on bounded domains.

The absorbing boundary conditions (ABC) for the wave equation was first introduced by Engquist and Majda [36]. The ABCs are designed so that they are local operators and they minimize the amount of artificial reflections. Bayliss and

Turkel characterized the ABCs as expansions of $1/r$ where r is the distance from the origin [8]. The idea was generalized to arbitrary smooth surfaces in [3]. The simplest approximation is the half-order ABC. Consider the basic unbounded problem for the Helmholtz equation given in Equation (2.7). We truncate the domain with a sphere of radius r. The truncated problem is expressed as

$$\begin{cases} \Delta u + k^2 u = 0 & x \in \Omega \\ \beta_1 u + \partial_{\mathbf{n}} u = 0 & x \in \Gamma \\ \partial_{\mathbf{n}} u - iku = 0 & \text{on } |x| = r, \end{cases} \tag{2.11}$$

The BVP thus has a finite computational domain $\Omega = |x| < r - S$, where S is the region inside of the scatterer. The lowest order ABC introduces some artificial reflections thus introducing error at the continuous level. The half order ABC will be used in many problems in this dissertation because the main goal is to study the domain decomposition.

The condition on the absorbing layer can also iterate together with the DDM. This technique is referred to as the adaptive radiation condition (ARC) [1, 11]. The ARC also has the benefit of truncating the problem to a finite domain. The right-hand side to the ARC is treated as an unknown along with the DDM iterations. The ARC is designed so that the right-hand side converges to the exact radiation condition, hence eliminating any artificial reflection error. This topic will be discussed further in the next chapter.

Another truncation technique is the perfectly matched layer (PML) originally proposed by Berenger [13]. ABCs are posed on a layer of zero thickness, and imposing the fully absorbing condition with high accuracy may be expensive and difficult to implement. The idea behind the PML is to enforce the absorption over a thin band. A simple PDE is imposed in the band such that no artificial reflections are introduced in the approximated solution.

2.2 Domain Decomposition Formulations

This section introduces several domain decomposition formulations. At the continuous level, domain decomposition reformulates a single BVP as an iterative sequence of local BVPs.

2.2.1 Direct Method for the Nonhomogeneous Problem

The nonhomogeneous problem given in Equation (2.10) is challenging because it contains a mix of an unbounded domain and a nonhomogeneous PDE. There is a direct method for solving this problem which involves linking the nonhomogeneous bilinear form with the integral equation for the exterior problem [60]. Assume an integral solution for the exterior PDE of the form

$$u_0(x) = u^{\text{inc}}(x) + Vp(x) + Nu(x), \quad x \in \Omega_0 \tag{2.12}$$

where Vp and Nu are the usual single- and double-layer potentials, respectively, associated with densities p and $u|_\Sigma$

$$Vp(x) := \int_\Sigma G_k(x,y)p(y)\mathrm{d}\Sigma_y, \quad Nu(x) := -\int_\Sigma \partial_{\mathbf{n}_0(y)} G_k(x,y)u(y)\mathrm{d}\Sigma_y \tag{2.13}$$

and $G_k(x,y)$ is the Green kernel for the Helmholtz in 2D with wavenumber k. The density p is an auxiliary unknown

$$p := \partial_{\mathbf{n}_0} u_0|_\Sigma = -\varepsilon^{-1}\partial_{\mathbf{n}} u\big|_\Sigma\,. \tag{2.14}$$

Substituting equation (2.12) into the Helmholtz and applying the standard jump conditions will yield an integral equation for the unknowns p and $u|_\Sigma$:

$$-\left(\frac{1}{2} - N\right) u(x) + Vp(x) = -u^{\text{inc}}(x) \quad \text{for } x \in \Sigma \tag{2.15}$$

The variable coefficient Helmholtz can be put into variational form to get a system in terms of p, $u|_\Sigma$, and $u|_\Omega$:

$$\int_\Omega \varepsilon^{-1} \left(\nabla u \cdot \nabla v - k^2 n^2 u v \right) \mathrm{d}\Omega + \int_\Sigma p v \mathrm{d}\Sigma = 0 \tag{2.16}$$

After suitable FEM and BEM discretizations, equations (2.15) and (2.16) yield a single linear system. Its stiffness matrix contains both sparse and dense blocks, complicating parallel solution. High frequencies and three-dimensional models further increase the system size, runtime, and memory requirements. In addition, some boundary integral formulations admit spurious modes. Domain decomposition can mitigate these difficulties [17]. The next section introduces a continuous domain decomposition algorithm and explains its advantages over direct coupling.

2.2.2 DDM for the Heterogeneous Problem

Before going to the full DDM with N subdomains, we will first consider the interior-exterior decomposition and discuss its advantages. First, rewrite the transmission conditions in Equation (2.9) as the following:

$$\begin{aligned} \beta u_0 + \partial_{\mathbf{n_0}} u_0 &= \beta u - \varepsilon^{-1} \partial_{\mathbf{n}} u \\ \beta u + \varepsilon^{-1} \partial_{\mathbf{n}} u &= \beta u_0 - \partial_{\mathbf{n_0}} u_0. \end{aligned} \tag{2.17}$$

Doing so, we introduce the transmission operator β acting on the continuity constraint posed on the artificial interface Σ. It is an important operator that governs the convergence properties of NDDMs and will be discussed further in a later section. Next, we define auxiliary functions

$$\begin{aligned} g_0(u) &= \beta u - \varepsilon^{-1} \partial_{\mathbf{n}} u \\ g(u_0) &= \beta u_0 - \partial_{\mathbf{n_0}} u_0 \end{aligned} \tag{2.18}$$

which are simply the right hand sides in Equation (2.17). Then we write the BVP in Equation (2.10) as two separate systems:

$$\mathfrak{F}_{\text{int}}(u;g) = \begin{cases} \nabla \cdot \left(\frac{1}{\varepsilon}\nabla u\right) + k^2 \frac{n^2}{\varepsilon} u = f_\Omega & x \in \Omega \\ \varepsilon^{-1}\partial_{\mathbf{n}} u = f_\Gamma & x \in \Gamma \\ \beta u + \varepsilon^{-1}\partial_{\mathbf{n}} u = g & x \in \Sigma, \end{cases} \tag{2.19}$$

$$\mathfrak{F}_{\text{ext}}(u_0;g_0) = \begin{cases} \Delta u_0 + k^2 u_0 = 0 & x \in \Omega_0 \\ |x|^{1/2}\left(\nabla u_0 \cdot \frac{x}{|x|} - iku_0\right) = 0 & |x| \to +\infty \\ \beta u_0 + \partial_{\mathbf{n_0}} u_0 = g_0 & x \in \Sigma. \end{cases} \tag{2.20}$$

The notation $\mathfrak{F}(u;g)$ denotes a BVP with unknown u and data g. From this point onward, we suppress the cumbersome ^{sc} superscript: u and u_0 denote the scattered wave amplitudes.

The problems in equations (2.19) and (2.20) are individually easier to solve than Equation (2.10). The finite element method is well suited to a variable-coefficient Helmholtz equation on a bounded domain, while a boundary integral method can handle the exterior homogeneous problem. However, the boundary data g and g_0 are unknown. NDDM solves the two problems independently with provisional data, then uses the auxiliary relations in Equation (2.18) to update each problem's boundary data from its neighbor's solution. Repeating this process produces an iterative algorithm:

Algorithm 2 separates the sparse interior FEM system $\mathfrak{F}_{\text{int}}(u;g)$ from the possibly dense exterior system $\mathfrak{F}_{\text{ext}}(u_0;g_0)$. Each can use a suitable linear solver, and both can be solved simultaneously before exchanging boundary data. Robin

Algorithm 2: Interior-Exterior NDDM. This is a basic iteration with fixed number of iterations.

Initialize $g^{(0)}$ and $g_0^{(0)}$;
Choose number of iterations M;
for $n = 1 : M$ **do**
  Solve $\mathfrak{F}_{\text{int}}(u^{(n)}; g^{(n-1)})$;
  Solve $\mathfrak{F}_{\text{ext}}(u_0^{(n)}; g_0^{(n-1)})$;
  Set $g^{(n)} = g(u_0^{(n)})$;
  Set $g_0^{(n)} = g_0(u^{(n)})$;
end

non-overlapping DDM also admits the efficient update

$$\begin{aligned} g_0^{(n)} &= 2\beta u^{(n)} - g^{(n-1)} \\ g^{(n)} &= 2\beta u_0^{(n)} - g_0^{(n-1)} \end{aligned} \tag{2.21}$$

This formula follows from Equations (2.17) and (2.18). This formula implies that the update step of the DDM iteration can avoid the expensive normal derivative approximation cost. This algorithm can be generalized to a further parallelized algorithm.

2.2.3 Non-overlapping DDM with Robin Conditions

Domain decomposition methods may use overlapping or non-overlapping subdomains. Non-overlapping methods avoid duplicate solutions in overlapping regions. Overlapping methods can use Dirichlet interface conditions and are often simpler to implement. For the Helmholtz problems in this dissertation, we consider non-overlapping methods.

Now we introduce the N-subdomain DDM. We take the interior domain Ω and split it into N disjoint subdomains Ω_i, $i = 1, \cdots, N$ such that:

- $\bar{\Omega} = \bigcup_{i=1}^{N} \bar{\Omega}_i, \quad i = 1, \cdots, N$

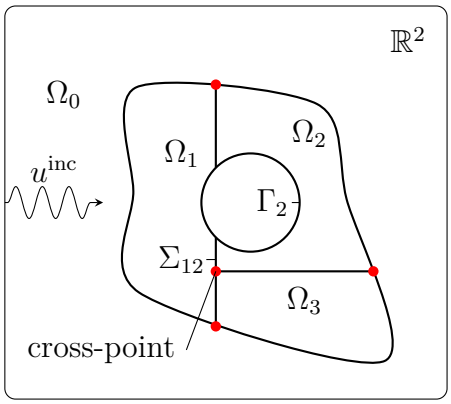


Figure 2.3 N-Subdomain DDM. The cross-points are shown in red.

- $\Omega_i \cap \Omega_j = \emptyset, \quad \forall i \neq j, \quad i, j = 1, \cdots, N$
- $\partial\Omega_i \cap \partial\Omega_j = \bar{\Sigma}_{ij} = \bar{\Sigma}_{ji}, \quad i, j = 1, \cdots, N$ is the artificial interface separating Ω_i from Ω_j

These conditions ensure that our subdomain configuration is regular, or conforming. For the sake of analysis and implementation, all artificial interfaces must separate exactly two bounded subdomains.

Figure 2.3 shows a three-subdomain configuration and highlights the cross-points, which may be shared by more than two subdomains or may meet Ω_0. Normal derivatives require special treatment at these points at the discrete level, as discussed below. We introduce the following notation:

- u_i is the interior solution to the original BVP restricted to domain Ω_i
- $\mathbf{n}_i$ is the unit normal vector pointing away from Ω_i
- $\Gamma_i = \Gamma \cap \bar{\Omega}_i$, and $\Sigma_{i0} = \Sigma \cap \bar{\Omega}_i$
- ε_i, n_i, $f_{\Omega,i}$, and $f_{\Gamma,i}$ are their respective restrictions to $\bar{\Omega}_i$
- $\mathbf{n}_i$ is the unit normal pointing out of Ω_i
- Λ_i is the set of all indices of subdomains adjacent to Ω_i

We define the smaller BVP on each subdomain as follows:

$$\mathfrak{F}_{\text{int},i}(u_i; g_{ij}; g_{i0}) = \begin{cases} \nabla \cdot \left(\dfrac{1}{\varepsilon_i} \nabla u_i \right) + k^2 \dfrac{n_i^2}{\varepsilon_i} u_i = f_{\Omega_i} & x \in \Omega_i \\ \varepsilon_i^{-1} \partial_{\mathbf{n}_i} u_i = f_{\Gamma,i} & x \in \Gamma_i \\ \beta_{\text{ext}} u_i + \varepsilon_i^{-1} \partial_{\mathbf{n}_i} u_i = g_{i0} & x \in \Sigma_{i0} \\ \beta_{\text{int}} u_i + \varepsilon_i^{-1} \partial_{\mathbf{n}_i} u_i = g_{ij} & x \in \Sigma_{ij} : j \in \Lambda_i. \end{cases} \tag{2.22}$$

Subdomains in general do not need to share a boundary with the scatterer or the exterior surface. In such cases Γ_i or Σ_{i0} are empty sets, and the conditions imposed on these boundaries are void. The transmission operator β along the exterior interface and the artificial interface is differentiated because it may be beneficial to use different operators. We define the residual function at the n^{th} iteration as the change in the boundary data

$$r^{(n)} = \sum_{i=1}^{N} \left(g_{i0}^{(n)} - g_{i0}^{(n-1)} + \sum_{j \in \Lambda_i} g_{ij}^{(n)} - g_{ij}^{(n-1)} \right). \tag{2.23}$$

We define the residual norm as $R^{(n)} = \|r^{(n)}\|$. A small residual is used as an indicator of convergence, although a universally appropriate stopping criterion is difficult to choose. A simple criterion stops the iteration when the residual falls below a prescribed threshold. The continuous N-subdomain algorithm is:

Corners require additional care. Normal derivative conditions on a physical boundary, such as the scatterer or coating surface, assume sufficient smoothness. Polygonal boundaries have corners where a unique normal direction is unavailable. This issue arises even without domain decomposition and requires special treatment in nodal finite elements, as discussed below.

The name N-subdomain NDDM counts the interior regions; including Ω_0 gives $N + 1$ computational subdomains. The efficient update Equation (2.21) remains available. The local solves in lines 5–7 can be performed largely in parallel, followed by

Algorithm 3: N-subdomain NDDM. Each interior problem and the exterior problem may all be solved concurrently.

Initialize all $g_{ij}^{(0)}$;
Initialize iteration $n = 1$, residual $R = 1$;
Choose a tolerance ϵ;
while $R > \epsilon$ **do**
  for $i = 1 : N$ **do**
    Solve $\mathfrak{F}_{\text{int},i}(u_i; g_{ij}^{(n-1)}; g_{i0}^{(n-1)})$;
  end
  Solve $\mathfrak{F}_{\text{ext}}(u_0^{(n)}; g_{0i}^{(n-1)})$;
  Calculate all updates $g_{ij}^{(n)} = 2\beta u_{ji}^{(n)} - g_{ji}^{(n-1)}$;
  Calculate $R = \|r^{(n)}\|$;
  n++;
end

the cross-point coupling step. Alternatively, the exterior solve $\mathfrak{F}_{\text{ext}}$ can be omitted by imposing an approximate condition on the outer boundary, using $\mathfrak{F}_{\text{int},i}(u_i; g_{ij}^{(n-1)}; 0)$ in line 8. The bounded heterogeneous problems are well suited to FEM. The distributed formulation is also useful when the full system exceeds the memory of a single node. In practice, Algorithm 3 requires acceleration; the next section describes a Krylov approach.

2.2.4 Domain Decomposition as a Preconditioner

The direct application of the Additive Schwarz method to a BVP leads to an iterative method in which the *local solutions* progressively converge. We may alternatively think of the Schwarz method as an iteration of the *boundary data* on the artificial interfaces. Indeed, if the boundary data converge $g^{(n)} \to g$ then so will the local solutions, and vice versa.

Consider an operator $\mathcal{A}$ such that an NDDM iteration can be expressed in terms of the boundary data:

$$g^{(n+1)} = \mathcal{A}g^{(n)} + b. \tag{2.24}$$

Here g collects the boundary data on all interfaces, and b contains the fixed contributions from the incident wave. Assuming a fixed point exists, convergence can be studied through the spectral properties of $\mathcal{I} - \mathcal{A}$, where $\mathcal{I}$ is the identity. The standard Schwarz iteration Equation (2.24) is a Jacobi fixed-point iteration for

$$(\mathcal{I} - \mathcal{A})g = b. \tag{2.25}$$

Thus the standard application of the Schwarz method is sometimes referred to as the Jacobi method. The spectral properties of the operator $\mathcal{I} - \mathcal{A}$ are often studied for specific cases to prove convergence.

The iteration operator $\mathcal{A}$ combines linear local solves with exchange of interface data. The local solves involve inverses of subdomain matrices, while the exchange reorders interface data. Because Equation (2.25) is linear, it can be solved with a Krylov subspace method instead of the Jacobi iteration Equation (2.24).

One option is to use ORTHODIR [59]. Let $A = (\mathcal{I} - \mathcal{A})$. The algorithm is given in Algorithm 4. ORTHODIR is the main Krylov method used in numerical implementations of this dissertation. Based on observations, ORTHODIR converges faster than the Jacobi and converges for a larger range of wavenumbers. Note that our update formula is $g_{ij}^{(n)} = 2\beta u_{ji}^{(n)} - g_{ji}^{(n-1)}$. Hence the spectral properties of A depend heavily on the transmission operator β. Several choices for these operators are discussed in the next section.

2.3 Transmission Operators

The transmission operator β acts on the continuity relation in the interface conditions Equation (2.17). Wave DDMs are often analyzed by separating propagating and evanescent modes [2, 15]. Modal analysis can diagonalize the relevant operators and reduce the convergence analysis to the error in each mode.

Algorithm 4: ORTHODIR method to solve $Ag = b$.

Initialize g_0, $r_0 = b - Ag_0$, and $p_0 = r_0$;
Choose a tolerance ϵ;
Let $j = 0$;
while $|r_j| > \epsilon$ **do**
  $a_j = \frac{(r_j, Ap_j)}{Ap_j, Ap_j}$;
  $g_{j+1} = g_j + \alpha_j Ap_j$;
  $r_{j+1} = r_j - \alpha_j Ap_j$;
  for $i = 1 : j$ **do**
    $\beta_{ij} = -\frac{(A^2 p_j, Ap_i)}{(Ap_i, Ap_i)}$;
  end
  $p_{j+1} = Ap_j + \sum_{i=0}^{j} \beta_{ij} p_i$;
  j++;
end

2.3.1 Examples of Transmission Operators

There are several options for the transmission operators. These operators are tailored for the Helmholtz equation. Some examples are listed here.

- $\beta^0 u = -iku$. This was the original choice proposed by Bruno Després in 1991 based on a low-order approximation of the Dirichlet to Neumann (DtN) operator [16]. It was shown in [15] that the resulting iteration operator only acts on the part of the spectrum corresponding to propagating modes while the eigenvalues related to the evanescent modes have unit modulus. Convergence can be proven for this operator but it is too slow to use in practice.

- $\beta^\chi u = -iku + \chi u$. This is the operator used in the evanescent modes damping algorithm [15] where χ is a self-adjoint positive operator. This transmission operator was specifically designed to damp the evanescent modes. They are as simple to implement as the Després operator but yields much faster convergence in practice.

- $\beta^{OO0} u = au$ and $\beta^{OO2} u = au - b\Delta_S u$. These are the generic optimized order zero and order two transmission operators. Δ_S is the Laplace-Beltrami operator on a surface S. a, b are complex numbers obtained by solving a min-max optimization problem on the rate of convergence [42].

- $\beta^{\mathrm{sq},\varepsilon} u = -ik\sqrt{1 + \mathrm{div}_S\left(\frac{1}{k_\varepsilon^2}\nabla_S\right)}u$. This is the square root transmission operator proposed in [16]. div_S represents the surface divergence of a tangent vector

field on S, and ∇_S represents the tangential gradient of the surface field. The operator has a parameter function ϵ which may depend on the surface S, and $k_\epsilon = k + i\epsilon$ is its corresponding complexified wavenumber. Note that if ϵ was chosen to be a scalar then we have: $-ik\sqrt{1 + k_\varepsilon^{-2}\Delta_S}u$. It is important to choose $\epsilon > 0$ for the well-posedness of the DDM algorithm. The symbol $\sqrt{z}$ designates the principal determination of the square-root of a complex number z with branch-cut along the negative real axis.

- $\beta^{N,\varepsilon} = \partial_n N(k + i\epsilon)$, where the notation $N(\kappa)$ represents the double-layer potential $Nu(x) := -\int_\Sigma \partial_{\mathbf{n}_0(y)} G_\kappa(x, y)u(y)$ with a Green's function $G_\kappa(x, y)$ which has a wavenumber κ. Like the square root, this integral equation operator contains a complexified wavenumber $k+i\epsilon$ for $\varepsilon \in \mathbb{R}^+$. This operator is an exact DtN map for the exterior Helmholtz problem.

These operators differ in convergence speed, implementation effort, and memory requirements. Discretizing the nonlocal integral operator produces a dense complex matrix, whereas EMDA can be represented by a complex scalar. The integral operator can yield faster convergence, but at a greater cost per iteration. Operators with additional parameters may perform well after tuning. These tradeoffs mean that the most suitable operator depends on the problem.

2.3.2 Dual Transmission Operator

Non-overlapping DDMs usually assign a single transmission operator to each interface. For scattering in a medium with refractive index n, the effective wavenumber is kn, which can be used to select the operator. If an interface separates media with different refractive indices, we may instead use two operators, one for each side. We refer to this choice as a dual transmission operator.

In the nonhomogeneous scattering problem, the scatterer may have different wave properties in each of its decomposed subdomains. Consider a case where the scatter is composed of N_d regions with a constant index of refraction n_i and constant relative permittivity ε_i in each region for $i = 1, \ldots, N_d$. Figure 2.4 shows an example of a scatterer decomposed into $N_d = 3$ layers of varying indices of refraction. The

governing PDE in each subdomain is given by

$$\frac{1}{\varepsilon_i}\Delta u + \frac{k^2 n_i^2}{\varepsilon_i} u = f_i \quad x \in \Omega_i \tag{2.26}$$

This type of problem is most suitably handled by NDDM. Typically, the same transmission operator is enforced on each of the interfaces. This is done usually out of convenience. Instead, two different operators may be enforced on one shared surface coming from each side of the interface. The motivation for this technique is that transmission operators typically depend on the wavenumber k and in this problem the domains on either side of an interface have different wave properties. We refer to this technique as the dual transmission operator.

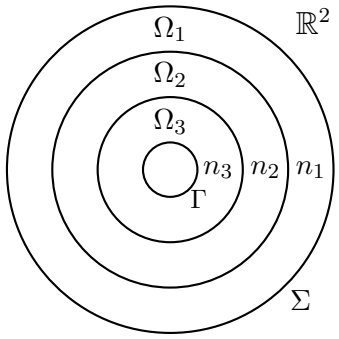


Figure 2.4 A multilayered scatterer with constant index of refraction in each layer.

Consider an artificial interface Σ_{ij} separating Ω_i and Ω_j. On Σ_{ij}, two different transmission operators T_i and T_j depending on n_i and n_j will be imposed on each side of the interface. The continuity equation will be written as

$$\begin{aligned} T_i u_i + \varepsilon_i^{-1} \partial_{\mathbf{n_i}} u_i &= T_i u_j - \varepsilon_j^{-1} \partial_{\mathbf{n_j}} u_j \\ T_j u_j + \varepsilon_j^{-1} \partial_{\mathbf{n_j}} u_j &= T_j u_i - \varepsilon_i^{-1} \partial_{\mathbf{n_i}} u_i. \end{aligned} \tag{2.27}$$

The right-hand sides of Equation (2.27) will be defined as g_{ij} and g_{ji}. Doing so allows us to write the local DDM problem as

$$\begin{cases} \dfrac{1}{\varepsilon_i}\Delta u_i^{(m+1)} + \dfrac{k^2 n_i^2}{\varepsilon_i} u_i^{(m+1)} = f_i x \in \Omega_i & \\ \varepsilon_i^{-1}\partial_{\mathbf{n_i}} u_i^{(m+1)} = g_i & x \in \Gamma_i \\ -iku_i^{(m+1)} + \varepsilon_i^{-1}\partial_{\mathbf{n}_i} u_i^{(m+1)} = g_{i0} & x \in \Sigma_{i0} \\ T_i u_i^{(m+1)} + \varepsilon_i^{-1}\partial_{\mathbf{n_i}} u_i^{(m+1)} = g_{ij}^{(m)} & x \in \Sigma_{ij}, \end{cases} \tag{2.28}$$

The update formula can be derived as follows:

$$\begin{aligned} g_{ij}^{(m+1)} &:= T_i u_j^{(m+1)} - \varepsilon_j^{-1}\partial_{\mathbf{n_j}} u_j^{(m+1)} \\ &= T_i u_j^{(m+1)} - (-T_i u_j^{(m+1)} + g_{ji}^{(m)}) \\ &= -g_{ji}^{(m)} + (T_i + T_j) u_j^{(m+1)}. \end{aligned} \tag{2.29}$$

The dual transmission algorithm is now defined; the remaining question is how to choose its operators. Both EMDA and the Padé operator depend on the wavenumber and can be written as $\beta = \beta(k)$. One choice is $T_i = \beta(kn_j)$ and $T_j = \beta(kn_i)$. The motivation for this approach is developed in Chapter 3.

2.4 Domain Decomposition at the Discrete Level

The preceding sections described three ways to replace the unbounded scattering problem by a bounded computation: absorbing boundary conditions, perfectly matched layers, and coupling to boundary integral equations. We now combine non-overlapping domain decomposition with the finite element method (FEM) using the continuous cross-point algorithm [10]. This algorithm underlies the contributions in Chapter 3. For a comprehensive introduction to FEM, see Gockenbach [44].

2.4.1 Weak Form

Before discretization, we express the PDE in weak form. A weak formulation relaxes the regularity required of the solution. Here, the formulation has the form $a(u, v) = \ell(v)$ and provides the basis for the finite element discretization.

Consider the nonhomogeneous interior problem Equation (2.22). We multiply the PDE by a test function, integrate over the domain, apply integration by parts, and incorporate the boundary conditions. For $v_i \in H^1(\Omega_i)$, the resulting weak form is

$$\int_{\Omega_i} \frac{1}{\varepsilon_i} \left(\nabla u_i \cdot \nabla v_i - k^2 n_i^2 u_i v_i \right) \mathrm{d}\Omega_i - ik \int_{\Sigma_{i0}} u_i v_0 \mathrm{d}\Sigma_{i0} + \sum_{j \in \Lambda_i} \int_{\Sigma_{ij}} \beta_{\mathrm{int}} u_i v_i \mathrm{d}\Sigma_{ij} = \int_{\Omega_i} f_{\Omega,i} v_i \mathrm{d}\Omega_i + \int_{\Sigma_{i0}} g_{i0} v_0 \mathrm{d}\Sigma_{i0} + \sum_{j \in \Lambda_i} \int_{\Sigma_{ij}} g_{ij} v_i \mathrm{d}\Sigma_{ij} + \int_{\Gamma_i} f_{\Gamma,i} v_i \mathrm{d}\Sigma_{i0} \quad (2.30)$$

A walkthrough of a simpler problem is given in Appendix A.3. The left-hand side defines a bilinear operator $a(u_i, v_i)$, and the right-hand side defines a linear operator $\ell(v_i)$. In the *strong form* the solution belongs to $C^2(\Omega_i)$ but the solution is $u_i \in H^1(\Omega_i)$ such that for any $v_i \in H^1(\Omega)$: $a(u_i, v_i) = \ell(v_i)$. The derivative constraint has been weakened from C^2 to H^1. For any space $\Omega \subset \mathbb{R}^N$, H^1 is the **Sobolev space** defined by [44]:

$$H^1(\Omega) := \left\{ v \in L^2(\Omega) \,:\, \frac{\partial v}{\partial x_i} \in L^2(\Omega), \forall i = 1, \ldots, n \right\}. \quad (2.31)$$

The space $L^2(\Omega)$ consists of the square integrable functions given by:

$$L^2(\Omega) := \left\{ v : \Omega \to \mathbb{R} \,:\, \int_\Omega v^2 < \infty \right\}. \quad (2.32)$$

The linear system for the FEM can now be defined. The bilinear operator $a(\cdot, \cdot)$ is actually an inner product. Therefore, the solution belonging to a finite dimensional subspace must have a constructable basis. Consider a general bilinear problem $a(w^h, v^h) = \ell(v)$ where w^h and v^h belong to an inner product space of

dimension n (cumbersome notation is dropped). The best approximation of $u \in C^2$ with the function w^h of the form

$$w = \sum_{i=1}^{n} \alpha_i w_i^h, \tag{2.33}$$

can be found by solving the linear system

$$G\alpha = b \tag{2.34}$$

where G is matrix with elements $G_{ij} = a(w_i^h, w_j^h)$ and b is a vector with elements $b_i = \ell(w_i^h)$. This technique is referred to as the Galerkin method for a variational problem [44]. The Galerkin method also applies to the BEM. Suppose w^h is now the BEM solution. Suppose the integral equation was of the form $Au = f$. Then the Galerkin system will be

$$\sum_{i=1}^{n} (w_j^h, Aw_i^h)\alpha_i = (w_j^h, f), \quad \text{for } k = 1, \ldots, n. \tag{2.35}$$

The bilinear operator here $(\cdot, \cdot)$ denotes the L^2 inner product over the exterior curve Σ, or more explicitly, $(f, g) := \int_\Sigma \overline{f(x)} g(x) ds(x)$.

The typical choice for the basis functions w_i are hat functions. Consider a domain that is meshed into n points located at $x_i \in \Omega, i = 1, \ldots, n$ (one for each hat function). Then the hat functions must obey the constraint:

$$w_i(x_j) = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \tag{2.36}$$

Values between mesh nodes are determined by interpolation. We use piecewise linear functions throughout this dissertation; higher-degree piecewise polynomials are also possible.

Constructing the matrix in Equation (2.35) is called assembly. The algorithm loops over elements and adds their contributions to the global matrix and load vector.

For each subdomain Ω_i, the usual FEM procedure constructs the local system. Further details are given in [44].

2.4.2 The Mesh

A finite element mesh partitions the computational domain into elements whose vertices are mesh nodes. The basis functions are defined by interpolation within each element. In a conforming two-dimensional mesh, distinct elements intersect only along a shared edge or vertex, or do not intersect.

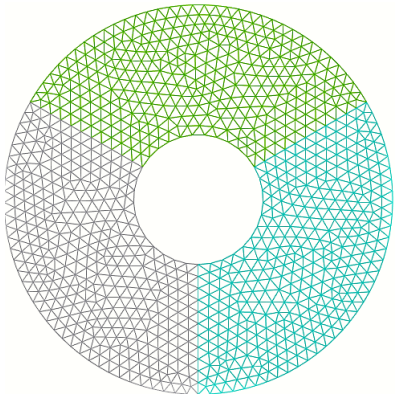


Figure 2.5 Example of a NDDM mesh with triangular elements.

In 2D, the typical choice of elements for FEM are triangles and quadrilaterals. Figure 2.5 shows a mesh with triangular elements. When a domain is meshed, the boundary mesh is naturally induced. In 2D, the elements for the BEM are simply edges.

To create a DDM mesh, an outline of the domain must be provided. The outline must have the list of coordinates of all the vertices and a list of the edges which connect the vertices. The edges must form closed areas which serve as the computational domains. Finally, the type of each edge must be classified as either natural or artificial. For the scattering problem, the natural edges must further be distinguished as either an exterior surface or a scattering surface. Equipped with this information, a DDM-FEM mesh may be generated. Each subdomain of a DDM-FEM

mesh has nodes and edges enumerated locally just like a standard FEM mesh. A DDM-FEM mesh provides additional information about the nodes that are shared between multiple subdomains. Each shared node must provide all domains with their neighbors' domain index and their local indexing for the node. Details about the mesh generation process is out of the scope of this dissertation.

2.4.3 Continuous Cross-point Algorithm

We now describe the continuous cross-point technique introduced in [10]. In a decomposed FEM mesh, corners of the subdomain boundaries $\partial\Omega_i$ may not have a uniquely defined normal direction. The algorithm replaces the interface derivative conditions at these nodes with a global continuity constraint.

We let $\mathcal{T}_i^h$ be the mesh on each subdomain and let X_i^h be the standard finite dimensional space of $\mathbb{P}_1$-continuous functions which approximates $H^1(\Omega_i)$. Every node on the mesh $\mathcal{T}_i^h$ can be classified as one of the following:

- Private - belonging to only 1 FEM subdomain. These nodes are interior to Ω_i, Σ_{i0}, and Γ_i. We may call them I-nodes.
- Shared - belonging to exactly 2 FEM subdomains. These nodes are interior to Σ_{ij}. We may call them S-nodes.
- Cross-point - corner vertices of the subdomains. For 2D meshes, these corners are equivalent to the points where any of the curves Γ_i, Σ_{ij}, or Σ_{i0} meet. These points belong to the boundary of the boundary of a set in R^d, and hence the set of cross-points in 2D is zero dimensional. We call them C-nodes.

Any function $v_i^h \in X_i^h$ can be decomposed as follows:

$$v_i^h = v_{iI}^h + \sum_{j\in\Lambda_i} v_{ij}^h + v_c^h, \tag{2.37}$$

where v_{iI}^h is supported on private nodes, v_{ij}^h on shared nodes, and v_c^h on cross-points. The usual global FEM space X^h consists of continuous piecewise polynomials over Ω.

We introduce the broken space X_B^h, defined as the span of functions of the form

$$v^h = \sum_{i=1}^{N} \left(v_{iI}^h + \sum_{j \in \Lambda_i} v_{ij}^h \right) + v_c^{h_c}. \tag{2.38}$$

The space X_B^h is the approximation space for the non-overlapping Robin DDM with continuous cross-points. Continuity is relaxed at shared nodes and enforced globally at cross-points. The discrete system is

$$\left.\begin{array}{l} a_i\left(u_{iI}^h + \sum_{\ell \in \Lambda_i} u_{i\ell}^h + u_c^h, v_{iI}^h\right) = L_i v_{iI}^h \quad \forall v_{iI}^h \in X_i^h \\ a_i\left(u_{iI}^h + \sum_{\ell \in \Lambda_i} u_{i\ell}^h + u_c^h, v_{ij}^h\right) + c_{ij}^h\left(\beta u_{ij}^h, v_{ij}^h\right) \\ \quad = L_i v_{ij}^h + c_{ij}^h\left(g_{ij}, v_{ij}^h\right) \quad \forall v_{ij}^h \in X_i^h, \quad j \in \Lambda_i \end{array}\right\}, \ i = 1, \ldots, N \tag{2.39}$$

$$\begin{array}{c} \sum_{i=1}^N a_i\left(u_{iI}^h + \sum_{\ell \in \Lambda_i} u_{i\ell}^h + u_c^h, v_c^h\right) = \sum_{i=1}^N L_i v_c^h \quad \forall v_c^h \in X_c^h \\ g_{ij} = -g_{ji} + 2\beta \, u_{ji}^h\big|_{\Sigma_{ij}} \end{array} \tag{2.40}$$

where a_i and $c_{ij}^h = c_{ji}^h$ are suitable bilinear forms expressing the variational form of the PDE in Ω_i and traces of u_{ij}^h and u_{ji}^h on Σ_{ij} respectively. This system generates the FEM linear system of the form:

$$\begin{bmatrix} A_{11} & & & & A_{1c} \\ & A_{22} & & & A_{2c} \\ & & \ddots & & \vdots \\ & & & A_{NN} & A_{Nc} \\ A_{c1} & A_{c2} & \ldots & A_{cN} & A_{cc} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \\ u_c \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_N \\ g_c \end{bmatrix} \tag{2.41}$$

where the vectors u_i represent the solution on the i^{th} domain, $u_{iI}^h + \sum_{\ell \in \Lambda_i} u_{i\ell}^h$. All of the sub blocks A_{ii}, A_{ic}, A_{ci}, and A_{cc} are sparse matrices. The index c denotes the cross-point system. The vector u_c corresponds to the nodal values on the cross-points. The A_{cc} matrix is much smaller compared to the A_{ii} systems due to the ratio of interior and interface nodes to cross-points. Although each of the systems A_{ii} are coupled at

the cross-points, a Schur complement procedure yields an efficient parallel algorithm to solve the system. If we express u_i as

$$u_i = (A_{ii})^{-1} (g_i - A_{ic} u_c), \tag{2.42}$$

then we can derive the following system in terms of u_c:

$$\left(A_{cc} - \sum_{i=1}^{N} A_{ci} (A_{ii})^{-1} A_{ic} \right) u_c = g_c - \sum_{i=1}^{N} A_{ci} (A_{ii})^{-1} g_i. \tag{2.43}$$

Let us define the left-hand and right-hand side of Equation (2.43) as A_{schur} and g_{schur} respectively. Although the matrix A_{schur} is dense, it is still relatively small. The following algorithm describes how to efficiently solve the system in Equation (2.41) for multiple iterations:

Algorithm 5: Schur Complement procedure for the FEM system

```
Do LU factorization on A_ii, ∀i;
LU-solve to compute (A_ii)^{-1} A_ic, ∀i;
Compute and do LU factorization on A_schur;
for each iteration do
    LU-solve to compute (A_ii)^{-1} A_ic, ∀i;
    Compute g_schur;
    LU-solve to compute u_c;
    Compute u_i;
end
```

Because the algorithm repeatedly solves systems with different right-hand sides, we use LU factorization. For a dense system of size m, factorization with pivoting costs approximately $(2/3)m^3$ operations, followed by $2m^2$ operations for permutation and triangular solves [31]. Factoring the sparse matrices A_{ii} is generally much cheaper. The cross-point system adds work relative to an uncoupled DDM, but improves convergence.

The operator β in this algorithm is scalar. The convergence proof in [10] applies to scalar transmission operators. More sophisticated operators can be implemented

within the framework, but that proof does not establish their convergence. Chapter 3 investigates several such extensions.

CHAPTER 3

CONTRIBUTIONS TO HELMHOLTZ DDM

This chapter presents contributions to Helmholtz DDM algorithms built on the continuous cross-point framework for nodal finite elements [10]. The topics include Padé and integral equation transmission operators, perfectly matched layers, dual transmission operators for heterogeneous media, adaptive radiation conditions, and an extension of cross-point continuity.

3.1 Padé-localized Square Root Operator

This section adapts the cross-point NDDM algorithm to the Padé-localized square-root transmission operator. Its effectiveness for the Helmholtz equation was established in [16]. We describe its implementation within the cross-point framework and assess its performance numerically.

3.1.1 Padé Approximation of the Square Root Operator

The square-root transmission operator is a nonlocal approximation of a DtN map. Its direct discretization produces a dense matrix, with greater memory and computational costs than a local operator. A Padé approximant provides a local rational approximation [54]. The order-N_p approximation to the complex square root has the form

$$\sqrt{1+z} \approx R^{\alpha}_{N_p}(z) = C_0 + \sum_{\ell=1}^{N_p} \frac{A_\ell z}{1 + B_\ell z}, \tag{3.1}$$

where z represents an operator, and C_0, A_ℓ, and B_ℓ are complex coefficients given by

$$\begin{aligned} C_0 =& e^{\mathrm{i}\frac{\alpha}{2}} R_{N_p}\left(e^{-\mathrm{i}\alpha} - 1\right) \\ A_\ell =& \frac{e^{-\frac{\mathrm{i}\alpha}{2}} a_\ell}{\left(1 + b_\ell \left(e^{-\mathrm{i}\alpha} - 1\right)\right)^2} \\ B_\ell =& \frac{e^{-\mathrm{i}\alpha} b_\ell}{1 + b_\ell \left(e^{-\mathrm{i}\alpha} - 1\right)}, \end{aligned} \tag{3.2}$$

where α is the angle of rotation. The a_ℓ, b_ℓ for $\ell = 1, \cdots, N_p$ are standard real Padé coefficients given by

$$\begin{aligned} a_\ell =& \frac{2}{2N_p + 1} \sin^2\left(\frac{\ell\pi}{2N_p + 1}\right) \\ b_\ell =& \cos^2\left(\frac{\ell\pi}{2N_p + 1}\right), \end{aligned} \tag{3.3}$$

which are used to approximate $\sqrt{1+x}$. We define the function

$$R_{N_p}(z) := 1 + \sum_{\ell=1}^{N_p} \frac{a_\ell z}{1 + b_\ell z} \tag{3.4}$$

which may take a complex argument and is used to compute C_0. It is possible to use $R_{N_p}(z)$ to directly approximate the square-root operator. The introduction of the complex coefficients is rooted in the optimization technique for Padé approximations. A change of variables involving α was used to derive the new complex coefficients C_0, A_ℓ, B_ℓ. The change of variables is a branch cut rotation which improves the convergence of the series [54].

The localized square-root operator contains an inverse. The $\frac{1}{1+B_\ell z}$ in Equation (3.1) symbolizes an inverse operation. To deal with that, we introduce an auxiliary unknown ϕ_ℓ at each order $\ell = 1, \cdots, N_p$. We will refer to this as the Padé-localized

square-root (PLS) operator:

$$\beta^{N_p,\alpha,\varepsilon}u = -\mathrm{i}k\left(C_0 u + \sum_{\ell=1}^{N_p} A_\ell \,\mathrm{div}_S\left(\frac{1}{k_\varepsilon^2}\nabla_S\varphi_\ell\right)\right), \tag{3.5}$$

where ϕ_ℓ for $\ell = 1, \cdots, N_p$ satisfy

$$\left(1 + B_\ell \,\mathrm{div}_S\left(\frac{1}{k_\varepsilon^2}\nabla_S\right)\right)\varphi_\ell = u. \tag{3.6}$$

The next section will discuss the discrete form of the PLS DDM algorithm with continuous cross-points.

3.1.2 Padé Operator with Cross-points

The cross-point algorithm in [10] uses a scalar transmission operator β. We now extend it to the PLS operator. Introducing auxiliary unknowns on the artificial interfaces produces an augmented linear system that can be solved by a Schur complement procedure, as in the scalar case.

Suppose we set $\beta_{\mathrm{ext}}u = -\mathrm{i}ku$ and $\beta_{\mathrm{int}}u = \beta^{N_p,\alpha,\varepsilon}u$. Then the variational form on Ω_i for the nonhomogeneous problem changes to:

$$\begin{aligned}\int_{\Omega_i}\frac{1}{\varepsilon_i}\left(\nabla u_i\cdot\nabla v_i - k^2 n_i^2 u_i v_i\right)\mathrm{d}\Omega_i - \mathrm{i}k\int_{\Sigma_{i0}} u_i v_0 \mathrm{d}\Sigma_{i0} - \mathrm{i}kC_0\sum_{j\in\Lambda_i}\int_{\Sigma_{ij}} u_i v_i \mathrm{d}\Sigma_{ij} \\ + \mathrm{i}k\sum_{j\in\Lambda_i}\sum_{\ell=1}^{N_p} A_\ell\int_{\Sigma_{ij}}\frac{1}{k_\varepsilon^2}\nabla_S\varphi_{i,\ell}\cdot\nabla_S v_i\mathrm{d}\Sigma_{ij} = \\ \int_{\Omega_i} f_{\Omega,i}v_i\mathrm{d}\Omega_i + \int_{\Sigma_{i0}} g_{i0}v_0\mathrm{d}\Sigma_{i0} + \sum_{j\in\Lambda_i}\int_{\Sigma_{ij}} g_{ij}v_i\mathrm{d}\Sigma_{ij} + \int_{\Gamma_i} f_{\Gamma,i}v_i\mathrm{d}\Sigma_{i0}\end{aligned} \tag{3.7}$$

which is coupled to the auxiliary system for $\ell = 1, \ldots, N_p$:

$$\sum_{j\in\Lambda_i}\left(\int_{\Sigma_{ij}} u_i v_{i,\ell}\mathrm{d}\Sigma_{ij} + B_\ell\int_{\Sigma_{ij}}\frac{1}{k_\varepsilon^2}\nabla_S\varphi_{i,\ell}\cdot\nabla_S v_{i,\ell}\mathrm{d}\Sigma_{ij} - \int_{\Sigma_{ij}}\varphi_{i,\ell}v_{i,\ell}\mathrm{d}\Sigma_{ij}\right) = 0. \tag{3.8}$$

Then write the discrete variational form as:

$$\left.\begin{array}{l} a_i\left(u_{iI}^h + \sum_{j\in\Lambda_i} u_{ij}^h + u_c^h, v_{iI}^h\right) = L_i(v_{iI}^h), \quad \forall v_{iI}^h \in X_i^h \\ a_i\left(u_{iI}^h + \sum_{j\in\Lambda_i} u_{ij}^h + u_c^h, v_{ij}^h\right) + \sum_{\ell=1}^{N_p} p_i(\phi_{i,\ell}^h, v_{ij}^h) = L_i(v_{ij}^h), \\ q_i(\sum_{j\in\Lambda_i} u_{ij}^h, v_{ij}^h) + r_i(\phi_{i,\ell}^h, v_{ij}^h) = 0, \quad \forall \ell = 1, \ldots, N_p, \\ \forall v_{ij}^h \in X_i^h, \quad j \in \Lambda_i \end{array}\right\}, i = 1, \ldots, N_d \tag{3.9}$$

$$\sum_{i=1}^{N} a_i\left(u_{iI}^h + \sum_{j\in\Lambda_i} u_{ij}^h + u_c^h, v_c^h\right) = \sum_{i=1}^{N} L_i(v_c^h) \quad \forall v_c^h \in X_c^h$$

$$g_{ij} = -g_{ji} + 2\beta^{N_p,\alpha,\varepsilon}\, u_{ji}^h\big|_{\Sigma_{ij}}.$$

These equations can be written more concisely as a linear system. Applying the Galerkin method yields the block system:

$$\begin{bmatrix} A_{11} & P_1 & & & & A_{1c} \\ Q_1 & R_1 & & & & 0 \\ & & \ddots & & & \vdots \\ & & & A_{NN} & P_N & A_{Nc} \\ & & & Q_N & R_N & 0 \\ A_{c1} & 0 & \ldots & A_{cN} & 0 & A_{cc} \end{bmatrix} \begin{bmatrix} u_1 \\ \varphi_1 \\ \vdots \\ u_N \\ \varphi_N \\ u_c \end{bmatrix} = \begin{bmatrix} g_1 \\ 0 \\ \vdots \\ g_N \\ 0 \\ g_c \end{bmatrix} \tag{3.10}$$

Notice that this system is an augmented version of the regular scalar system found in Equation (2.41). Each standard FEM subblock A_{ii} is augmented by three additional matrices P_i, Q_i, and R_i. The submatrix P_i is constructed by discretizing the bilinear

operator

$$a_P(\varphi_{i,1},\ldots,\varphi_{i,N_p};v_i) = \mathrm{i}k\sum_{j\in\Lambda_i}\sum_{\ell=1}^{N_p} A_\ell \int_{\Sigma_{ij}} \frac{1}{k_\varepsilon^2}\nabla_S\varphi_{i,\ell}\cdot\nabla_S v_i \mathrm{d}\Sigma_{ij} \tag{3.11}$$

from Equation (3.7). The submatrices Q_i and R_i are constructed by discretizing all orders $\ell = 1,\ldots,N_p$

$$a_Q(u_i,v_{i,\ell}) = \sum_{j\in\Lambda_i}\left(\int_{\Sigma_{ij}} u_i v_{i,\ell}\mathrm{d}\Sigma_{ij}\right) \tag{3.12}$$

and

$$a_R(\varphi_{i,\ell},v_{i,\ell}) = \sum_{j\in\Lambda_i}\left(B_\ell\int_{\Sigma_{ij}}\frac{1}{k_\varepsilon^2}\nabla_S\varphi_{i,\ell}\cdot\nabla_S v_{i,\ell}\mathrm{d}\Sigma_{ij} - \int_{\Sigma_{ij}}\varphi_{i,\ell}v_{i,\ell}\mathrm{d}\Sigma_{ij}\right) \tag{3.13}$$

from Equation (3.8) respectively. Since the right-hand side of Equation (3.8) is zero, the RHS to the linear system is simply padded with zeros. The vector φ_i contains the nodal values of the auxiliary unknowns corresponding to $\varphi_{i,1},\ldots,\varphi_{i,N_p}$. The size of φ_i is equal to the number of shared nodes on $\partial\Omega_i$ multiplied by the Padé order N_p.

A Schur complement procedure can be applied to the PLS linear system, Equation (3.10), just as it was applied to scalar system. If we write φ_i as

$$\varphi_i = -(R_i)^{-1}Q_i u_i \tag{3.14}$$

then we obtain the following expression for u_i independent of ϕ_i

$$\begin{aligned}
A_{ii}u_i + P_i\varphi_i + A_{ic}u_c &= g_i \\
A_{ii}u_i + P_i\left(-(R_i)^{-1}Q_i u_i\right) + A_{ic}u_c &= g_i \\
\left(A_{ii} - P_i(R_i)^{-1}Q_i\right)u_i &= g_i - A_{ic}u_c \\
\implies u_i = \left(A_{ii} - P_i(R_i)^{-1}Q_i\right)^{-1}&\left(g_i - A_{ic}u_c\right)
\end{aligned} \tag{3.15}$$

Notice that the new expression for u_i is almost the same as in Equation (2.42). That means we can reuse Algorithm 4 from Subsection 2.2.4 by making a small

modification. Let us define

$$\tilde{A}_{ii} = A_{ii} - P_i(R_i)^{-1}Q_i \tag{3.16}$$

which is the matrix being inverted in Equation (3.15). The first step in the initialization is to construct $\tilde{A}_{ii}$. Then we can follow Algorithm 4 except we replace A_{ii} in the recipe with $\tilde{A}_{ii}$. We can use Equation (3.14) as a formula to compute φ_i after u_i has been computed. This step must be added to Algorithm 4 because we need the auxiliary solution to perform the update. The variational forms of the update formulas are

$$\int_{\Sigma_{i0}} g_{i0}^{(n)} v_0 = -2\mathrm{i}k \int_{\Sigma_{i0}} u_0^{(n)} v_0 - \int_{\Sigma_{i0}} g_{0i}^{(n-1)} v_0 \tag{3.17}$$

on the exterior interface, and

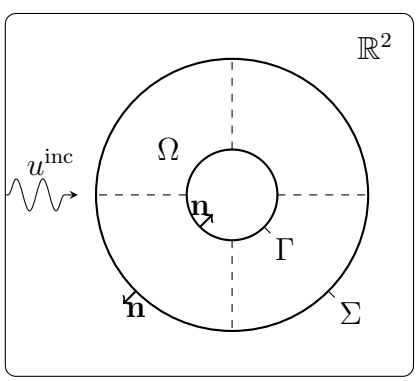
$$\int_{\Sigma_{ij}} g_{ij}^{(n)} v_i = -2\mathrm{i}kC_0 \int_{\Sigma_{ji}} u_{ji}^{(n)} v_i - 2\mathrm{i}k \sum_{\ell=1}^{N_p} A_\ell \int_{\Sigma_{ji}} \frac{1}{k_\varepsilon^2} \nabla_S \varphi_{j,\ell}^{(n)} \cdot \nabla_S v_i - \int_{\Sigma_{ji}} g_{ji}^{(n-1)} v_i \tag{3.18}$$

on each artificial interface $j \in \Lambda_i$. As in other FEM-DDMs, a new load vector is assembled at each iteration. The terms involving g_{ij} need not be assembled from scratch: $g_{ij}^{(0)}$ is initialized from the incident-field contributions involving f_Γ and f_Ω, and subsequent updates require only the terms involving $u_j^{(n)}$ and $\varphi_{i,\ell}^{(n)}$. Load-vector assembly has linear cost. In parallel, each subdomain assembles its neighbors' contributions from its local solution, after which the vectors are exchanged. This update is inexpensive relative to the linear solves.

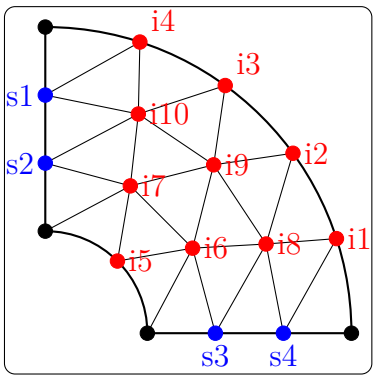
3.1.3 Detailed Example

Consider an annulus Ω centered at the origin, with inner boundary Γ representing the scatterer and outer boundary Σ truncating the exterior problem. Figure 3.1(a) shows this geometry, whose physical boundaries have no corners. We partition it radially into four equal subdomains and introduce a coarse mesh on one subdomain, shown

in Figure 3.1(b). This example illustrates the nodal FEM-NDDM implementation of the Padé operator.



(a) Simple case for a scattering problem with a truncated exterior domain.



(b) A subdomain of geometry on the left with mesh nodes.

Figure 3.1 Supplementary figures to demonstrate the implementation of the PLS operator assembly.

The mesh in Figure 3.1(b) contains 18 nodes, classified as I-nodes (red), S-nodes (blue), and C-nodes (black). The degree-of-freedom indices determine their positions in the solution vectors. Let ddl_i and ddl_s count the I- and S-nodes, respectively. The local vector u_1 has dimension $ddl_i + ddl_s$=10+4=14. The figure labels identify nodal coordinates in counterclockwise order. The first ddl_i entries correspond to I-nodes, and entries $ddl_i + 1$ through $ddl_i + ddl_s$ correspond to S-nodes. The auxiliary vector φ_1 has dimension $ddl_s \times N_p$, ordered by the same S-node indices. For $N_p = 1$, the

vectors are

$$u_1 = \begin{bmatrix} u(i1) \\ u(i2) \\ \vdots \\ u(i10) \\ u(s1) \\ u(s2) \\ u(s3) \\ u(s4) \end{bmatrix} \qquad \varphi_1 = \begin{bmatrix} \varphi_1(s1) \\ \varphi_1(s2) \\ \varphi_1(s3) \\ \varphi_1(s4) \end{bmatrix}. \tag{3.19}$$

In this example, the auxiliary unknown has dimension $ddl_s = 4$. Equation (3.19) expresses u_1 and φ_1 in terms of nodal values. The auxiliary function φ_1 is supported on artificial interfaces and excludes corners. The local finite element system on Ω_1 is

$$\begin{bmatrix} A_{11} & P_1 \\ Q_1 & R_1 \end{bmatrix} \begin{bmatrix} u_1 \\ \varphi_1 \end{bmatrix} = \begin{bmatrix} g_1 \\ 0 \end{bmatrix}. \tag{3.20}$$

We focus on a local subsystem of Equation (3.10), temporarily setting aside the corner system. The matrices A_{11}, P_1, Q_1, and R_1 are **stiffness matrices**, and g_1 is the **load vector**. Assembly populates these matrices and vectors by approximating the integrals in the weak formulation. For the Padé transmission operator, this formulation is

$$a(u_1, v) + p(\varphi_1, v) = \ell(v) \tag{3.21}$$

where

$$
\begin{aligned}
a(u_1, v) &= \int_{\Omega_1} \frac{1}{\varepsilon_1} \left(\nabla u_1 \cdot \nabla v - k^2 n_1^2 u_1 v \right) \mathrm{d}\Omega_1 - \mathrm{i}k \int_{\Sigma_{1,0}} u_1 v \, \mathrm{d}\Sigma_{1,0} - \mathrm{i}kC_0 \sum_{j \in \Lambda_i} \int_{\Sigma_{1,j}} u_1 v \, \mathrm{d}\Sigma_{1,j}, \\
p(\varphi_1, v) &= \mathrm{i}k \sum_{j \in \Lambda_i} A_1 \int_{\Sigma_{1,j}} \frac{1}{k_\varepsilon^2} \nabla_S \varphi_1 \cdot \nabla_S v \, d\Sigma_{1,j}, \\
\ell(v) &= \int_{\Omega_i} f_{\Omega,i} v_i \mathrm{d}\Omega_i + \int_{\Sigma_{i0}} g_{i0} v_0 \mathrm{d}\Sigma_{i0} + \sum_{j \in \Lambda_i} \int_{\Sigma_{ij}} g_{ij} v_i \mathrm{d}\Sigma_{ij} + \int_{\Gamma_i} f_{\Gamma,i} v_i \mathrm{d}\Sigma_{i0}.
\end{aligned}
\tag{3.22}
$$

Approximating $a(\cdot, \cdot)$ leads to the assembly of A_{11}, approximating $p(\cdot, \cdot)$ to P_1, and $\ell(\cdot)$ to g_1. The bilinear part $a(\cdot, \cdot) + p(\cdot, \cdot)$ consists of an area integral over Ω_1 and contour integrals over $\Sigma_{1,j}$. In Figure 3.1, we see the area Ω_1 approximated by a patch of triangles which naturally forms an approximation of the contours by a chain of edges. Algorithmically speaking, the assembly consists of looping over each element and approximating the integrals in the weak form restricted to that particular element. Then the quadrature values at the nodes of the element are added to their proper locations on the stiffness matrix. Looping over the triangles in Ω_i will add contributions to the A_{ii}. If a triangle contains a corner as a vertex, then A_{1c} and A_{c1} are updated.

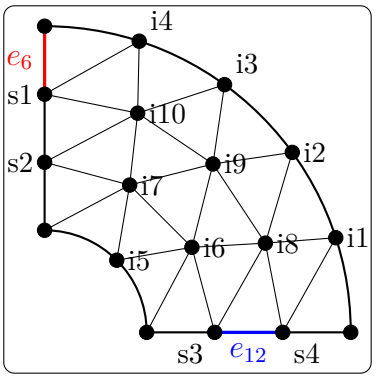


Figure 3.2 Shows the edges e_6 marked in blue and e_{12} marked in blue.

We loop over edges to approximate the contour integrals. A boundary edge can lie on the scatterer Γ_1, the exterior $\Sigma_{1,0}$, or on one of the artificial interfaces $\Sigma_{1,j}$. If the edge is on Γ_1 and the edge contains a corner as an endpoint, then the load vector g_c is updated accordingly. If the edge is on $\Sigma_{1,0}$ and contains a corner as an

endpoint, then matrices $A_{1,c}$ and $A_{c,1}$ are updated accordingly. However, when the edge lies on an artificial interface, the corner stiffness matrices will not be affected even when one of the endpoints is a corner. This is our treatment of our corners in the nodal FEM-NDDM. Corner nodes have their own system which links the nodes of all domains, but this system is only affected for corners that lie on the Σ or Γ. When we loop over edges on an artificial interface Σ_{ij}, there are no contributions to the corner system.

Suppose we are integrating over the edge e_6 on the exterior (refer to Figure 3.2). Since one of the endpoints of e_6 is a corner, the integral only results in one contribution to the A_{11} and P_1 each. The integral $-\mathrm{i}kC_0 \int_{e_6} u_1 v_i \mathrm{d}s$ contributes to the A_{11}, and $\mathrm{i}kA_1 \int_{e_6} \frac{1}{k_\xi^2} \nabla_S \varphi_1 \cdot \nabla_S v \, ds$ contributes to the P_1 matrix. The matrix in the left-hand side of Equation (3.20) can be expanded as below in Equation (3.23). Edge e_{12}, has 2 S-nodes for endpoints. The entries in the A and P which are affected are marked in blue in Equation (3.23).

$$\left(\begin{array}{ccc|cccc|cccc} a_{1,1} & \cdots & a_{1,10} & & & & & & & & \\ \vdots & \ddots & \vdots & & & & & & \mathbf{0} & & \\ a_{10,1} & \cdots & a_{10,10} & & & & & & & & \\ \hline & & & a_{11,11} & a_{11,12} & 0 & 0 & p_{11,1} & p_{11,2} & 0 & 0 \\ & & & a_{12,11} & a_{12,12} & 0 & 0 & p_{12,1} & p_{12,2} & 0 & 0 \\ & & & 0 & 0 & a_{13,13} & a_{13,14} & 0 & 0 & p_{13,3} & p_{13,4} \\ & & & 0 & 0 & a_{14,13} & a_{14,14} & 0 & 0 & p_{14,3} & p_{14,4} \\ \hline & Q_1 & & & & & & & R_1 & & \end{array}\right) \tag{3.23}$$

The Q_1 and R_1 matrices are assembled by approximating the bilinear form in (3.8).

3.1.4 Numerical Results for the Padé Operator

This section draws on the 2020 conference contribution by Boubendir and Takahashi on cross-points and Padé approximants for the Helmholtz equation, prepared for the volume *Domain Decomposition Methods in Science and Engineering XXVI* [19].

This section is devoted to validating the proposed method through numerical simulations. We generate a triangular mesh controlled by 2 parameters: wavelength λ and points per wavelength n_λ. We use both a Krylov subspace solver (ORTHODIR) and successive approximation (Jacobi) methods to solve the iteration operator [59]. We compare the algorithm based on Padé approximants [16] with the one called evanescent mode damping algorithm (EMDA) [14] which consists of choosing $\beta = -\mathrm{i}k(1 + i\mathcal{X})$. We evaluate the performance based on the number of iterations until *convergence* (initial residual decreases by a factor of 10^{-6}). For the Padé parameters, we chose [16] $\alpha = \pi/4$ and $\varepsilon = 0.6k^{1/3}$. For the EMDA, we chose $\mathcal{X} = 1/2$. These simulations are performed on the 2 geometries shown in Figure 3.3.

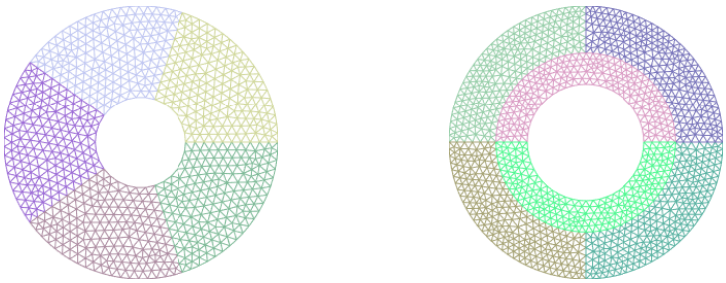


Figure 3.3 Geometries used for the numerical simulation: pie (left) and layered (right) configurations.

In the pie configuration (left of Figure 3.3), the annulus is split radially into N_d equal subdomains. The inner radius is fixed $r_1 = 1$ and the outer radius varies with frequency $r_2 = r_1 + 2\pi/k$. In the first experiment, we fixed $N_d = 5$, $k = \pi$, varied $n_\lambda =$ 12, 16, 20, 24, and Padé orders $N_p = 1$, 2, 4, 8. Table 3.1 shows the obtained results. The transmission operators are listed in the second row. The EMDA is denoted by E and the PLS operator of order N_p is denoted by P(N_p).

Table 3.1 Iteration Number with respect to n_λ

	Jacobi					ORTHODIR				
n_λ	E	P(1)	P(2)	P(4)	P(8)	E	P(1)	P(2)	P(4)	P(8)
12	105	33	25	25	25	28	18	15	14	14
16	136	44	28	24	24	32	20	17	14	14
20	159	52	34	23	23	35	22	18	15	15
24	192	63	42	23	23	38	24	20	16	15

In the second experiment, we determine how the wavenumber affects the convergence rate. Table 3.2 displays the results for fixed $N_d = 5$, $n_\lambda = 16$ and wavenumber varied $k = \pi$, 2π, 3π, 4π. In the third experiment, we varied the number of subdomains $N_d = 2$, 4, 6, 12 with fixed $k = \pi$ and $n_\lambda = 16$. These results are summarized in Table 3.3.

Table 3.2 Iteration Number with respect to k

	Jacobi					ORTHODIR				
k	E	P(1)	P(2)	P(4)	P(8)	E	P(1)	P(2)	P(4)	P(8)
π	136	44	28	24	24	32	20	17	14	14
2π	124	40	25	22	22	29	19	16	14	14
3π	127	41	26	24	24	30	20	16	15	15
4π	120	39	25	24	24	29	19	16	15	15

Table 3.3 Iteration Number with respect to N_d

	Jacobi					ORTHODIR				
N_d	E	P(1)	P(2)	P(4)	P(8)	E	P(1)	P(2)	P(4)	P(8)
2	136	44	28	17	17	20	14	13	12	11
4	137	44	27	20	20	29	18	15	13	13
6	139	44	28	25	25	32	20	17	15	15
12	139	67	54	50	49	39	28	24	23	23

The layered configuration (right of Fig. 3.3) is a six-domain annulus in which the radii $r_1 = 1, r_2 = 3$ and N_d are fixed. For this geometry, we first varied $n_\lambda = 12$, 16, 20, 24 with fixed $k = \pi$. We then tested several wavenumbers $k = \pi$, 2π, 3π, 4π with fixed $n_\lambda = 16$. All the obtained results are listed in Table 3.4.

Table 3.4 Iteration Numbers for the Layered Configuration

n_λ	Jacobi E	Jacobi P(2)	Jacobi P(4)	ORTHODIR E	ORTHODIR P(2)	ORTHODIR P(4)
12	110	60	59	36	24	24
16	123	70	68	40	25	25
20	144	76	73	43	27	26
24	165	81	79	46	29	27

k	Jacobi E	Jacobi P(2)	Jacobi P(4)	ORTHODIR E	ORTHODIR P(2)	ORTHODIR P(4)
π	123	70	68	40	25	25
2π	115	44	44	42	31	31
3π	110	53	53	45	37	37
4π	165	56	56	46	42	42

These tests demonstrate the effectiveness of combining the Padé-localized square-root operator with continuous cross-point treatment. The observed convergence is robust across the tested parameters, particularly with ORTHODIR.

3.2 Boundary Integral Equation Transmission Operator

This section adapts the cross-point NDDM to boundary integral equation (IE) transmission operators. Integral equations are often used to represent solutions on unbounded exterior domains. Here, they instead define nonlocal transmission operators intended to accelerate DDM convergence.

Dirichlet-to-Neumann (DtN) maps motivate effective transmission operators, including the square-root approximation discussed above. Hypersingular Helmholtz boundary integral operators provide another option. Steinbach discussed IE transmission operators in [62], and Boubendir et al. demonstrated their effectiveness in [18].

For the n-dimensional Helmholtz equation, the **single-** and **double-layer potentials** are defined by

$$\begin{aligned} Vp(x) &= \int_S G_n(x,y)p(y)dS_y, \\ N\lambda(x) &= -\int_S \partial_{\mathbf{n}_y} G_n(x,y)\lambda(y)dS_y, \end{aligned} \tag{3.24}$$

where S is a closed curve in $\mathbb{R}^n$ and $\mathbf{n}$ is its outward normal. For $x \neq y$, the n-dimensional Helmholtz **Green kernel** is

$$G_n(x,y) = \begin{cases} (i/4)H_0^{(1)}(k|x-y|), & \text{for } n=2, \\ \frac{\exp(ik|x-y|)}{4\pi|x-y|} & \text{for } n=3, \end{cases} \tag{3.25}$$

where $H_0^{(1)}$ is the **Hankel function** of the first kind of order 0, and k is the wavenumber. The derivative of the double-layer potential is a hypersingular operator defined in the distributional sense by

$$D_\kappa \lambda = \begin{cases} \nabla_S \cdot V\left(\nabla_S \lambda \times \mathbf{n}\right) \times \mathbf{n} - \kappa^2 \mathbf{n} \cdot V(\lambda \mathbf{n}), & n=3, \\ -\partial_s V\left(\partial_s \lambda\right) - \kappa^2 \mathbf{n} \cdot V(\lambda \mathbf{n}), & n=2, \end{cases} \tag{3.26}$$

where κ is the effective wavenumber, ∇_S is the surface gradient, and ∂_s is the unit-speed parameter on S. See [12] for more details about boundary integral operators for the Helmholtz equation.

3.2.1 DDM with IE Transmission Operator

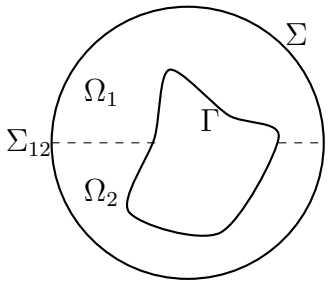


Figure 3.4 Sample geometry of a DDM with scatterer Γ and truncating surface Σ.

Consider a non-overlapping domain decomposition problem with a static condition on the scattering surface Γ and a standard absorbing boundary condition on the exterior surface Σ. On the artificial interface Σ_{ij}, the transmission operator will be an integral equation operator $\mathcal{T}^S$ where S is the closed curve defined for the

IE operator. The problem on the local domain Ω_i is expressed as

$$\begin{cases} (\Delta + k^2)u_i^{(n)} = 0 & x \in \Omega_i \\ (\beta + \partial_{\mathbf{n}_i})u_i^{(n)} = g^{\text{inc}} := -(\beta + \partial_{\mathbf{n}})u^{\text{inc}} & x \in \Gamma_i \\ \partial_{\mathbf{n}_i}u_i^{(n)} + \mathcal{T}^S u_i^{(n)} = g_{ij}^{(n-1)} & x \in \Sigma_{ij} : j \in \Lambda_i \\ (-\mathrm{i}k + \partial_{\mathbf{n}_i})u_i^{(n)} = 0 & x \in \Sigma_i \end{cases}, \quad \forall i. \tag{3.27}$$

Figure 3.4 shows the geometry. The curve S is a selectable parameter rather than an intrinsic boundary of the problem. Here (n) denotes the iteration index and β is the scalar impedance coefficient on the scatterer. The weak form of Equation (3.27) is

$$\int_{\Omega_i} (\nabla u_i^{(n)} \cdot \nabla v_i - k^2 u_i^{(n)} v_i)\mathrm{d}x + \int_{\Gamma_i} \beta u_i^{(n)} v_i \,\mathrm{d}s + \int_{\Sigma_{ij}} \mathcal{T}^S[u_i^{(n)}]v_i \mathrm{d}s = \int_{\Gamma_i} g^{\text{inc}} v_i \,\mathrm{d}s + \int_{\Sigma_{ij}} g_{ij}^{(n-1)} v_i \mathrm{d}s \tag{3.28}$$

Lastly, the weak form of the boundary update is given by

$$\int_{\Sigma_{ij}} g_{ij}^{(n+1)} v_j \,\mathrm{d}s = -\int_{\Sigma_{ij}} g_{ji}^{(n)} v_j \,\mathrm{d}s + 2\int_{\Sigma_{ij}} \mathcal{T}^S[u_j^{(n+1)}]v_j \,\mathrm{d}s. \tag{3.29}$$

Equations (3.28) and (3.29) give the bilinear formulation of the DDM. We use the FEM-DDM mesh described in Subsection 2.4.2 and apply the Galerkin method to obtain a system for the nodal values. Assembly combines standard FEM contributions with BEM contributions from $\mathcal{T}^S$. The resulting local systems contain large sparse blocks and smaller dense blocks.

We consider two options for the IE transmission operator $\mathcal{T}^S$. The first is

$$\mathcal{T}_0^S u := -\mathrm{i}ku + D_0 u. \tag{3.30}$$

This operator involves the hypersingular operator with wavenumber $\kappa = 0$. This corresponds to the DtN for a Laplace equation. The second operator is

$$\mathcal{T}_{\epsilon}^{S} u := 2D_{k+i\epsilon} u. \tag{3.31}$$

The second operator also involves the hypersingular operator but has a complexified wavenumber with a parameter $\epsilon > 0$. It has been shown that for smooth surfaces, the complexification of the wavenumber led to favorable properties for the numerical solution of IEs [21, 30].

The function $\mathcal{T}^{S} u(x)$ is defined for $x \in \Sigma_{ij}$ and will have singularities if $x = y$ for any $y \in S$. If S is chosen so that it does not intersect with Σ_{ij} then the operator will have no singularities.

3.2.2 Concentric Example

In this section, we consider a simple example where the artificial interface is a closed curve. Since Σ_{ij} is a closed curve, the choice of $S = \Sigma_{ij}$ is very appropriate. Secondly, we investigate the influence of the parameter ϵ on the convergence rate of the DDM. Lastly, we demonstrate the effectiveness of the IE operators compared to the Padé operator for various test cases.

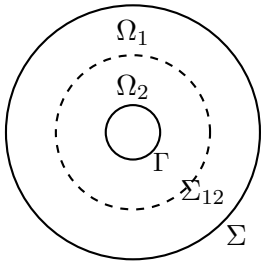


Figure 3.5 Two-subdomain geometry with $S = \Sigma_{ij}$ as the closed curve.

Consider the Scattering problem on the annulus. See Appendix A.2. The annulus can be sub-divided into concentric circles which creates artificial interfaces that are closed curves. See Figure 3.5. We may use the DDM Equation (3.27) and let

the closed curve for the hypersingular operator be the interface itself, i.e. $S = \Sigma_{ij}$. The inner radius is 1, the artificial interface radius is 2, the outer radius is 3. For the base test, let wavenumber $k = \pi$ and points-per-wavelength $n_\lambda = 16$. The two main experiments are:

- Vary $n_\lambda = 12$, 16, 20, 24
- Vary $k = \pi/2$, π, $3\pi/2$, 2π

The goal of the experiment is to study how the complexification parameter ϵ for the second IE operator responds to the change in the other parameters. Then we compare the performance of the optimized operator to the Laplace IE operator.

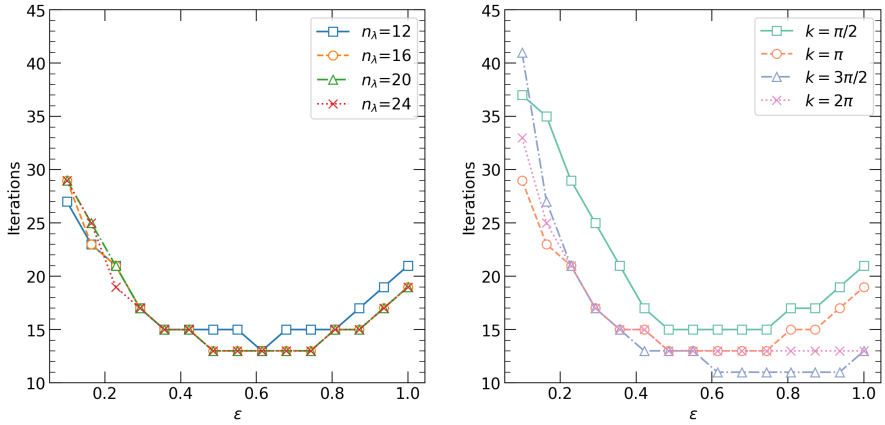
Figure 3.6 plots iteration counts as ϵ varies over a discretization of $(0, 1)$. The tested problems exhibit a range of values between ϵ_{min} and ϵ_{max} with minimal iteration counts for both Jacobi and ORTHODIR. The preferred range depends on the wavenumber, but is less sensitive to discretization: curves for different n_λ nearly coincide. This suggests that a coarser discretization may help select ϵ for a finer problem. Parameter selection matters: changing ϵ by 0.5 reduces one Jacobi count from 29 to 14, while the highest-wavenumber ORTHODIR test, $k = 2\pi$, decreases from 18 to 10 iterations.

Table 3.5 Iteration Numbers for n_λ and k Tests on the Concentric Domain

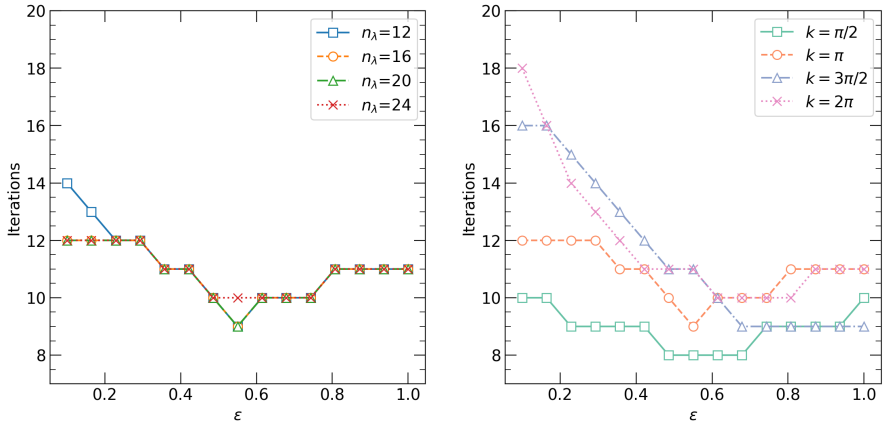
	Jacobi			ORTHODIR		
n_λ	P(4)	$\mathcal{T}_0$	$\mathcal{T}_\epsilon$	P(4)	$\mathcal{T}_0$	$\mathcal{T}_\epsilon$
12	19	77	13	15	17	10
16	18	77	13	15	17	10
20	17	79	13	15	16	10
24	17	79	13	15	16	10

	Jacobi			ORTHODIR		
k	P(4)	$\mathcal{T}_0$	$\mathcal{T}_\epsilon$	P(4)	$\mathcal{T}_0$	$\mathcal{T}_\epsilon$
$\pi/2$	19	21	15	11	12	8
π	18	77	13	15	17	10
$3\pi/2$	13	33	11	15	17	10
2π	15	200	13	17	21	10

Table 3.5 compares Padé with $N_p = 4$, the Laplace IE operator $\mathcal{T}_0$, and the complexified IE operator $\mathcal{T}_\epsilon$, using Jacobi and ORTHODIR iterations. The Padé complexification parameter is $\epsilon = -0.4k^{(1/3)}$ [16]; the second IE operator uses a



(a) n_λ test, Jacobi **(b)** k test, Jacobi



(c) n_λ test, ORTHODIR **(d)** k test, ORTHODIR

Figure 3.6 These plots show the iteration number of the complexified IE operator with respect to $\epsilon \in (0, 1)$ for the two-subdomain concentric geometry.

numerically optimized parameter. Iteration counts are relatively insensitive to n_λ. As k increases, ORTHODIR counts grow, while the Jacobi results are less regular. Among these tests, the optimized complexified IE operator converges fastest.

3.2.3 Adaptation of the IE Operator with Cross-points

The IE operator can be generalized to work on any artificial interface by using cutoff functions. Suppose a subdomain Ω_i has an interface Σ_{ij} which is not a closed curve. We let the closed curve for the double-layer be the boundary of Ω_i, i.e. $S = \partial\Omega_i$. Then use a smooth cutoff function $\chi_{\Sigma_{ij}}$ to restrict the value of the operator everywhere

outside of Σ_{ij}. The adapted DDM can be formally expressed as

$$\begin{cases} (\Delta + k^2)u_i^{(n)} = 0 & x \in \Omega_i \\ (\beta + \partial_{\mathbf{n}_i})u_i^{(n)} = g^{\text{inc}} := -(\beta + \partial_{\mathbf{n}})u^{\text{inc}} & x \in \Gamma_i \\ \partial_{\mathbf{n}_i}u_i^{(n)} + \chi_{\Sigma_{ij}}(\mathcal{T}^{\partial\Omega_i})u_i^{(n)} = g_{ij}^{(n-1)} & x \in \Sigma_{ij} : j \in \Lambda_i \\ (-\mathrm{i}k + \partial_{\mathbf{n}_i})u_i^{(n)} = 0 & x \in \Sigma_i \end{cases}, \quad \forall i. \tag{3.32}$$

This technique for IE transmission operators for general interfaces was introduced in [18]. At the discrete level there will be corners. This method can be used in tandem with the continuous cross-point technique for nodal finite elements.

3.2.4 Results for General Interfaces

We repeat the experiments from Subsection 3.1.4 with the two IE operators. Figure 3.3 shows the pie configuration, with a variable number of subdomains, and the layered configuration, with six subdomains. The baseline parameters are $k = \pi$, $n_\lambda = 16$, and $N_d = 5$ for the pie configuration. The three tests are:

- Vary $n_\lambda = 12$, 16, 20, 24
- Vary $k = \pi$, 2π, 3π, 4π
- Vary $= N_d$ 2, 4, 6, 12 for the pie configuration

The goal is to see how these parameters affect the convergence rate of the DDM. The iteration number for the Jacobi method and the preconditioned ORTHODIR are both presented. The parameter ϵ for the complexified IE operator is chosen in a way that minimizes the iteration number.

Tables 3.6 and 3.7 give the pie-configuration results. Iteration counts remain stable as n_λ varies. Compared with Table 3.1, both IE operators require fewer iterations than the order-eight Padé operator. In the k test, ORTHODIR counts

Table 3.6 Iteration Numbers for n_λ and k Tests for the Pie Domain

n_λ	Jacobi $\mathcal{T}_0$	Jacobi $\mathcal{T}_\epsilon$	ORTHODIR $\mathcal{T}_0$	ORTHODIR $\mathcal{T}_\epsilon$
12	22	19	14	10
16	22	18	14	10
20	22	17	14	10
24	23	17	14	11

k	Jacobi $\mathcal{T}_0$	Jacobi $\mathcal{T}_\epsilon$	ORTHODIR $\mathcal{T}_0$	ORTHODIR $\mathcal{T}_\epsilon$
π	22	18	14	10
2π	21	15	13	10
3π	36	17	13	10
4π	27	17	14	10

Table 3.7 N_d Test for the Pie Domain

N_d	Jacobi $\mathcal{T}_0$	Jacobi $\mathcal{T}_\epsilon$	ORTHODIR $\mathcal{T}_0$	ORTHODIR $\mathcal{T}_\epsilon$
2	18	14	11	8
4	18	14	12	9
6	30	22	15	12
8	62	27	17	14
12	2674	105	21	20

remain bounded over the tested range, as in Table 3.2. The Laplace IE and Padé operators perform similarly, while the complexified IE operator performs best. Counts increase with N_d in Table 3.7, although the IE operators improve on the Padé ORTHODIR results in Table 3.3. The unusually high Jacobi count for the Laplace operator at $N_d = 12$ indicates deteriorating convergence. The optimal ϵ for each of these test runs is 0.

Table 3.8 The n_λ and k Tests for the Layered Configuration

n_λ	Jacobi Lap.	Jacobi Comp.	ORTHODIR Lap.	ORTHODIR Comp.
12	DIV	288	23	17
16	DIV	93	24	18
20	DIV	64	25	18
24	DIV	51	24	18

k	Jacobi Lap.	Jacobi Comp.	ORTHODIR Lap.	ORTHODIR Comp.
π	DIV	93	24	18
2π	DIV	40	32	17
3π	DIV	26	36	18
4π	DIV	31	42	19

The results for two tests on the layered configuration are presented in Table 3.8. For both tests, the Laplace IE operator does not converge with the Jacobi iteration,

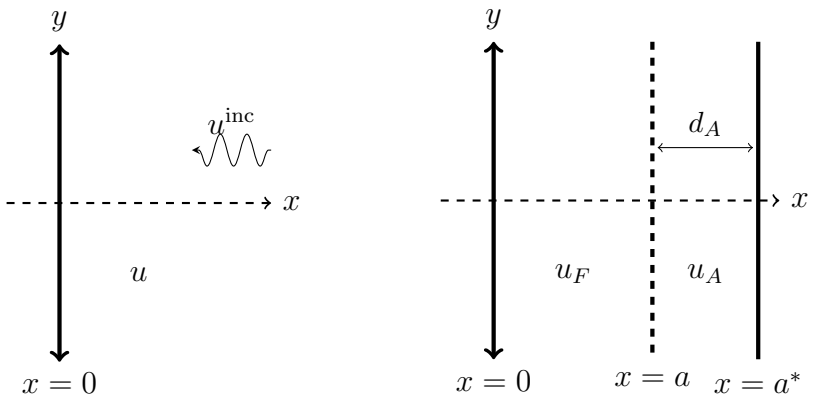
but converges as fast as the Padé operator for the ORTHODIR. The ORTHODIR methods show that their convergence is stable with respect to change in the discretization parameter n_λ. The Jacobi results for the n_λ suggests that the more refined mesh allows the complexified operator to converge faster. Again, $\epsilon = 0$ for the operator $\mathcal{T}_\epsilon$ for all n_λ. As for the k test, the ORTHODIR methods show a linear growth in iterations for the Laplace and a stable iterations for the complexified. In the k test, the optimal values of ϵ for $k = \pi$, 2π, 3π, 4π were 0.0, 0.7, 0.5, and 0.2 respectively.

These results demonstrate that cutoff IE operators can be used with continuous cross-point treatment. With ORTHODIR, the Laplace IE operator performs comparably to Padé, while the optimized complexified operator performs better. The optimal parameter ϵ_0 depends on the geometry and wavenumber, so selecting it a priori is difficult. A numerical parameter search over $[0, k/2]$ is a practical option for these examples.

These experiments demonstrate that IE transmission operators can accelerate DDMs on meshes with corners. With ORTHODIR, the Laplace IE operator performs comparably to Padé, while the optimized complexified IE operator performs better in the tested cases. Krylov acceleration is preferable to direct Jacobi iteration here. The main cost is the additional computation and memory required by dense interface matrices; this cost becomes more significant as the mesh surface-to-volume ratio increases.

3.3 Perfectly Matched Layer

The perfectly matched layer (PML), introduced by Bérenger [13], addresses the unbounded scattering domain by adding an absorbing shell of nonzero thickness. A modified PDE attenuates outgoing waves within the shell before an outer boundary



(a) Original model **(b)** Truncated model with PML

Figure 3.7 Sketch of the domain for the original problem Equation (3.33) and the truncated model Equation (3.34).

condition is imposed. This section applies the PLS operator with continuous cross-points to a PML problem.

PMLs may be curved or Cartesian. A Cartesian PML naturally creates multiple computational regions with corners, making it a useful application of the cross-point algorithm.

3.3.1 Convergence on the Half-plane

To motivate this application, we analyze a DDM for a half-plane scattering problem. The argument follows the square-root-operator analysis in Midura's dissertation [53]. Consider the two-dimensional Helmholtz problem

$$\begin{cases} \Delta u + k^2 u = 0 \quad \text{for } x > 0 \\ u(0, y) = f(y) := -u^{\text{inc}}(y) \\ \lim\limits_{r \to +\infty} \sqrt{r} \left(\dfrac{\partial u}{\partial r} - \mathrm{i}ku \right) = 0 \end{cases} \tag{3.33}$$

where u^{inc} is the incident wave and r is the polar coordinate. See Figure 3.7 (a) for a sketch of the domain.

We introduce a PML of width d_A in the vertical strip $a < x < a^* := a + d_A$ to truncate the unbounded domain in the x-direction. 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 $u_F = u|_{\Omega_F}$. See Figure 3.7 (b) for a sketch of the domain. The truncated problem with the PML is expressed as

$$\begin{cases} \Delta u_F + k^2 u_F = 0 \quad \text{for} \quad 0 < x < a \\ \gamma^{-1}\partial_x \left(\gamma^{-1}\partial_x u_A\right) + \partial_y^2 u_A + k^2 u_A = 0 \\ u_F(0, y) = f(y) \\ u_F(a, y) = u_A(a, y) \\ \partial_x u_F(a, y) = \gamma^{-1}(a)\partial_x u_A(a, y) \\ u_A\left(a^*, y\right) = 0, \end{cases} \tag{3.34}$$

where

$$\gamma(x) = \begin{cases} 1 & 0 < x < a \\ 1 + \frac{\mathrm{i}\sigma^*}{k}(x - a)^p & a \leq x < a^*. \end{cases} \tag{3.35}$$

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.

The continuity constraint on the interface Σ_{AF} is rewritten with a transmission operator β:

$$\begin{aligned} \partial_x u_F + \beta u_F &= \partial_x u_A + \beta u_A \\ -\partial_x u_A + \beta u_A &= -\partial_x u_F + \beta u_F. \end{aligned} \tag{3.36}$$

Doing so, the BVP can now be rewritten as the two decoupled systems:

$$\mathscr{P}_F\left\{u_F^{(n+1)}; f, g_F^{(n)}\right\} := \begin{cases} \Delta u_F^{(n+1)} + k^2 u_F^{(n+1)} = 0 \quad \text{for} \quad 0 < x < a \\ u_F^{(n+1)}(0, y) = f(y) \\ \partial_x u_F^{(n+1)}(a, y) + \beta u_F^{(n+1)}(a, y) = g_F^{(n)}(y) \end{cases} \tag{3.37}$$

and

$$\mathscr{P}_A\left\{u_A^{(n+1)}; g_A^{(n)}\right\} := \begin{cases} \gamma^{-1}\partial_x\left(\gamma^{-1}u_A^{(n+1)}\right) + \partial_y^2 u_A^{(n+1)} + k^2 u_A^{(n+1)} = 0 \\ u_A^{(n+1)}\left(a^*, y\right) = 0 \\ -\partial_x u_A^{(n+1)}(a,y) + \beta u_A^{(n+1)}(a,y) = g_A^{(n)}(y), \end{cases} \tag{3.38}$$

where

$$g_F^{(n)} = \partial_x u_A^{(n)} + \beta u_A^{(n)}, \quad \text{and} \quad g_A^{(n)} = -\partial_x u_F^{(n)} + \beta u_F^{(n)}. \tag{3.39}$$

The boundary data satisfy the following update formula

$$g_F^{(n+1)} = -g_A^{(n)} + 2\beta u_A^{(n+1)}, \quad \text{and} \quad g_A^{(n+1)} = -g_F^{(n)} + 2\beta u_F^{(n+1)}. \tag{3.40}$$

Equations (3.37, 3.38, 3.40) define the PML NDDM iteration for a general transmission operator β. To establish convergence, we show that the errors

$$\begin{aligned} e_F^{(n)} &= u_F^{(n)} - u_F \\ e_A^{(n)} &= u_A^{(n)} - u_A \end{aligned} \tag{3.41}$$

converge to zero as $n \to \infty$. By linearity, applying the NDDM to the error functions in Equation (3.41) gives

$$\begin{cases} \mathscr{P}_F\left\{e_F^{(n+1)}; 0,\, h_F^{(n)}\right\}, \\ \mathscr{P}_A\left\{e_A^{(n+1)}; h_A^{(n)}\right\}, \\ h_F^{(n+1)} = -h_A^{(n)} + 2\beta e_A^{(n+1)}, \\ h_A^{(n+1)} = -h_F^{(n)} + 2\beta e_F^{(n+1)}. \end{cases} \tag{3.42}$$

The precise definition of the error boundary data is given by

$$\begin{aligned} h_F^{(n)} &= g_F^{(n)} - \left(\partial_x u_F + \beta u_F\right)\big|_{x=a} \\ h_A^{(n)} &= g_A^{(n)} + \left(\partial_x u_A - \beta u_A\right)\big|_{x=a} \end{aligned} \tag{3.43}$$

If $h_F^{(n+1)}$ and $h_A^{(n+1)}$ converge to zero, the corresponding errors also vanish. We establish this by studying the Fourier transforms of the boundary data.

Definition 3.3.1. For any function of two variables $f(x,y) \in L^2$, we define its transform pair in the y variable to be

$$\begin{aligned} \hat{f}(x,\omega) = \mathcal{F}\{f\} &:= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x,y) e^{-\mathrm{i}\omega y} dy, \\ f(x,y) = \mathcal{F}^{-1}\{\hat{f}\} &:= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{f}(x,\omega) e^{\mathrm{i}\omega y} d\omega. \end{aligned} \tag{3.44}$$

The Fourier transform reduces the PDEs to ODEs and replaces the pseudodifferential operator $\mathcal{B}$ by its symbol. By Plancherel's theorem, convergence of $\hat{h}_F^{(n)}$ and $\hat{h}_A^{(n)}$ in the transformed norm implies convergence of $h_F^{(n)}$ and $h_A^{(n)}$. The following lemma gives the transformed iteration operator for a general β.

Lemma 3.3.1. *Consider the iteration scheme in Equation* (3.42). *Let* $\vec{\hat{h}}^{(n)} := [\hat{h}_F^{(n)}, \hat{h}_A^{(n)}]^T$. *Then the Fourier transform of the boundary data updates according to*

$$\hat{\mathbf{h}}^{(n+1)} = \hat{\mathbf{A}} \hat{\mathbf{h}}^{(n)}. \tag{3.45}$$

The iteration operator $\hat{\mathbf{A}}$ *has the following explicit form*

$$\hat{\mathbf{A}} = \begin{pmatrix} 0 & -1 + 2\sigma \frac{e^{\lambda a} - e^{2\lambda\xi(a^*)} e^{-\lambda a}}{(\sigma-\lambda)e^{\lambda a} - (\sigma+\lambda)e^{2\lambda\xi(a^*)}e^{-\lambda a}} \\ -1 + 2\sigma \frac{e^{\lambda a} - e^{-\lambda a}}{(\lambda+\sigma)e^{\lambda a} + (\lambda-\sigma)e^{-\lambda a}} & 0 \end{pmatrix} \tag{3.46}$$

where $\lambda = \mathrm{i}\sqrt{\omega^2 - k^2}$. *The* $\sigma(\omega)$ *represents the Fourier symbol of the transmission operator ie.* $\mathcal{B}e(x,y) = \mathcal{F}^{-1}\{\sigma(\omega)\hat{e}(x,\omega)\}$.

Proof. The stretched coordinate transform

$$\xi(x) = \int_a^x \gamma(s) ds = x + \frac{\mathrm{i}\sigma^*}{k} \int_a^x (s-a)^p ds, \tag{3.47}$$

can remove the anisotropic term from the PDE in Equation (3.38). The system of equations in the absorbing layer $\mathscr{P}_A \left\{ e_A^{(n+1)}; h_A^{(n)} \right\}$ after the change of variables

becomes

$$\begin{cases} \partial_\xi^2 E_A^{(n+1)} + \partial_y^2 E_A^{(n+1)} + k^2 E_A^{(n+1)} = 0 \quad \text{for} \quad \xi(a) < \xi < \xi\left(a^*\right) \\ E_A^{(n+1)}\left(\xi\left(a^*\right), y\right) = 0 \\ -\partial_\xi E^{(n+1)}(\xi(a), y) + \mathcal{B} E_A^{(n+1)}(\xi(a), y) = h_A^{(n)}(y) \end{cases} \tag{3.48}$$

where $E_A^{(n)}(\xi(x), y) = e_A^{(n)}(x, y)$. Next, we apply the Fourier transform to both systems to reduce them to ODEs. The transformation of $\mathscr{P}_F\left\{e_F^{(n+1)}; 0,\, h_F^{(n)}\right\}$ yields

$$\begin{cases} \partial_x^2 \hat{e}_F^{(n+1)} + \left(k^2 - \omega^2\right) \hat{e}_F^{(n+1)} = 0 \quad \text{for} \quad 0 < x < a \\ \hat{e}_F^{(n+1)}(0, \omega) = 0 \\ \partial_x \hat{e}_F^{(n+1)}(a, \omega) + \sigma(\omega) \hat{e}_F^{(n+1)}(a, \omega) = \hat{h}_F^{(n)}(\omega), \end{cases} \tag{3.49}$$

and similarly, the transformation of Equation (3.48) yields

$$\begin{cases} \partial_\xi^2 \hat{E}_A^{(n+1)} + \left(k^2 - \omega^2\right) \hat{E}_A^{(n+1)} = 0 \quad \text{for} \quad \xi(a) < \xi < \xi\left(a^*\right) \\ \hat{E}_A^{(n+1)}\left(\xi\left(a^*\right), \omega\right) = 0 \\ -\partial_\xi \hat{E}_A^{(n+1)}(\xi(a), \omega) + \sigma(\omega) \hat{E}_A^{(n+1)}(\xi(a), \omega) = \hat{h}_A^{(n)}(\omega). \end{cases} \tag{3.50}$$

Now we solve the systems. Both Equation (3.49) and Equation (3.50) can be solved with the ansatz of the form

$$u(x) = R(\omega) \exp\left(\mathrm{i}\sqrt{\omega^2 - k^2} x\right) + L(\omega) \exp\left(-\mathrm{i}\sqrt{\omega^2 - k^2} x\right). \tag{3.51}$$

We set $R(\omega) = 0$ in Equation (3.51) because our domain is the right half-plane and we do not seek a solution which grows as $x \to \infty$. The solutions to Equation (3.49) and Equation (3.50) respectively are

$$\begin{aligned} \hat{e}_F^{(n+1)} &= \frac{\hat{h}_F^{(n)}\left(\exp\left(\lambda x\right) + \exp\left(-\lambda x\right)\right)}{(\lambda + \sigma)\exp\left(\lambda a\right) + (\lambda - \sigma)\exp\left(-\lambda a\right)} \\ \hat{e}_A^{(n+1)} &= \frac{\hat{h}\left(\exp\left(\lambda(\omega)\xi(x)\right) + \exp\left(2\lambda\xi\left(a^*\right)\right)\exp\left(-\lambda(\omega)\xi(x)\right)\right)}{(\sigma - \lambda)\exp\left(\lambda a\right) - (\sigma + \lambda)\exp\left(2\lambda\xi\left(a^*\right)\right)\exp\left(-\lambda a\right)} \end{aligned} \tag{3.52}$$

where $\lambda = i\sqrt{\omega^2 - k^2}$. The solution of Equation (3.50) has been expressed in the original variable x. The Fourier transform of the update in Equation (3.42) is

$$\begin{aligned} \hat{h}_F^{(n+1)} &= -\hat{h}_A^{(n)} + 2\sigma(\omega)\hat{e}_A^{(n+1)} \\ \hat{h}_A^{(n+1)} &= -\hat{h}_F^{(n)} + 2\sigma(\omega)\hat{e}_F^{(n+1)} \end{aligned} \tag{3.53}$$

Substituting Equation (3.52) into Equation (3.53) and evaluating at $x = a$ recovers Equation (3.46). □

We have obtained the iteration operator in Fourier space. To relate it to the original NDDM, define the subblocks

$$\hat{\mathbf{A}}(\omega) = \begin{pmatrix} 0 & \hat{A}_{12}(\omega) \\ \hat{A}_{21}(\omega) & 0 \end{pmatrix}. \tag{3.54}$$

The iteration operator can be used to determine the convergence rate of the DDM. The next lemma proposes the convergence factor μ which is an upper bound for the spectral radius of $\hat{\mathbf{A}}(\omega)$. Indeed, this quantity actually depends on the choice of the transmission operator β. For now, we assume an admissible β exists.

Lemma 3.3.2. *Consider the PML NDDM iteration scheme given by Equations (3.37,3.38,3.40). Let* $\vec{h}^{(n)} := [h_F^{(n)}, h_A^{(n)}]^T$ *be the boundary data for the error of the DDM given in Equation* (3.42). *Let us define the quantity* $\mu = \max_{\omega\in\mathbb{R}} \left|\hat{A}_{12}(\omega)\hat{A}_{21}(\omega)\right|$. *If* $\mu < 1$ *then the Jacobi-type iteration scheme converges linearly i.e.*

$$\left\|\mathbf{h}^{(n+2)}\right\| \leq \mu \left\|\mathbf{h}^{(n)}\right\|. \tag{3.55}$$

Proof. Equation (3.45) with Equation (3.54) implies the following recurrence relation for $n \geq 0$:

$$\begin{aligned}\hat{h}_F^{(n+2)}(\omega) &= \hat{A}_{12}(\omega)\hat{h}_A^{(n+1)}(\omega) = \hat{A}_{12}(\omega)\hat{A}_{21}(\omega)\hat{h}_F^{(n)}(\omega),\\ \hat{h}_A^{(n+2)}(\omega) &= \hat{A}_{21}(\omega)\hat{h}_F^{(n+1)}(\omega) = \hat{A}_{21}(\omega)\hat{A}_{12}(\omega)\hat{h}_A^{(n)}(\omega).\end{aligned} \tag{3.56}$$

Another way of writing this is

$$\vec{\hat{h}}^{(n+2)} = \begin{pmatrix} \hat{A}_{12}(\omega)\hat{A}_{21}(\omega) & 0 \\ 0 & \hat{A}_{21}(\omega)\hat{A}_{12}(\omega) \end{pmatrix} \vec{\hat{h}}^{(n)}. \tag{3.57}$$

We will now use $L2$ norm to introduce inequalities. A standard matrix-vector product estimate yields

$$\left\|\vec{\hat{h}}^{(n+2)}\right\| \leq \left\| \begin{matrix} \hat{A}_{12}(\omega)\hat{A}_{21}(\omega) & 0 \\ 0 & \hat{A}_{21}(\omega)\hat{A}_{12}(\omega) \end{matrix} \right\| \left\|\vec{\hat{h}}^{(n)}\right\| \leq \mu \left\|\vec{\hat{h}}^{(n)}\right\|. \tag{3.58}$$

By the Plancherel theorem, we have the equivalence $\left\|\vec{\hat{h}}^{(n)}\right\| = \left\|\vec{h}^{(n)}\right\|$, which concludes the proof. □

The previous lemma has established the convergence rate for PML NDDM on the half plane in 2D for some admissible transmission operator β in L^2. We will finally prove that the complexified square root operator converges. In the context of the half-plane problem

$$\begin{aligned}\beta^{\mathrm{sq},\varepsilon} &= -\mathrm{i}k\sqrt{1 + \nabla_\Sigma \cdot \left(\frac{1}{k_\epsilon^2}\nabla_\Sigma\right)}\\ &= -\mathrm{i}k\sqrt{1 + \frac{1}{k_\varepsilon^2}\partial_x^2}.\end{aligned} \tag{3.59}$$

Since the interface $\Sigma = \{(a, y)|y \in \mathbb{R}\}$ is a straight line, it suffices to assume ε is independent of x. The Fourier symbol for this operator is

$$\sigma^{\mathrm{sq},\varepsilon}(\omega) = -\mathrm{i}k\sqrt{1 - \frac{\omega^2}{(k + \mathrm{i}\varepsilon)^2}}. \tag{3.60}$$

For simplicity, let us consider $\beta^{\mathrm{sq},0}$. Note that

$$\sigma^{\mathrm{sq},0}(\omega) = -\mathrm{i}\sqrt{k^2-\omega^2} = -\lambda. \tag{3.61}$$

The following theorem establishes the proof of the convergence with this transmission operator.

Theorem 3.3.3. *The PML NDDM iteration scheme given by Equation* (3.37)-*Equation* (3.38) *and Equation* (3.40) *for the half-plane converges if we choose the square-root transmission operator* $\beta^{sq,0}$.

Proof. To prove convergence, we examine the factor μ in Lemma 3.3.2. Substituting the symbol Equation (3.61) into Equation (3.46) gives

$$\begin{aligned} \hat{A}_{12}(\omega) &= -\exp\left(-2\lambda a\right)\exp\left(2\lambda\xi(a^*)\right), \\ \hat{A}_{21}(\omega) &= -\exp\left(2\lambda a\right). \end{aligned} \tag{3.62}$$

Therefore, we have

$$\begin{aligned} \mu &= \max_{\omega\in\mathbb{R}} \left|\hat{A}_{12}(\omega)\hat{A}_{21}(\omega)\right| \\ &= \max_{\omega\in\mathbb{R}} \left|\exp\left(2\lambda\xi(a^*)\right)\right| \\ &= \max_{\omega\in\mathbb{R}} \left|\exp\left(2\mathrm{i}\sqrt{k^2-\omega^2}\left(a^* + \frac{\mathrm{i}\sigma^* d^{p+1}}{k(p+1)}\right)\right)\right| \\ &\leq \max_{\omega\in\mathbb{R}} \left|\exp\left(2\mathrm{i}a^*\sqrt{k^2-\omega^2}\right)\right| \left|\exp\left(-2\sqrt{k^2-\omega^2}\frac{\sigma^* d^{p+1}}{k(p+1)}\right)\right|. \end{aligned} \tag{3.63}$$

Assume $k^2-\omega^2 > 0$. Then left exponential term is a pure oscillation, so it is bounded by 1. The right exponential term is an exponential decay since the argument is a real negative number. Therefore, $\mu < 1$. If $k^2-\omega^2 < 0$ then the roles of the left and right exponential terms are switched. Thus, we still have $\mu \leq 1, \forall\omega \in \mathbb{R}$. □

That concludes the DDM convergence proof for the PML problem on the half-plane with the complexified square root operator on the interface.

3.3.2 The Cartesian PML Problem

We now consider a scatterer centered at the origin and surrounded by a Cartesian PML. In the numerical tests, the scatterer is circular. The region Ω_F between the scatterer and the PML is the physical domain, with thickness parameter d_F, and retains the original PDE. The shell Ω_A uses a modified PDE to attenuate outgoing waves. The two regions are coupled across $\Sigma_{AF} := \overline{\Omega_A} \cap \overline{\Omega_F}$, and an outer condition is imposed on Σ. Figure 3.8 illustrates the geometry. We solve the coupled problem using NDDM.

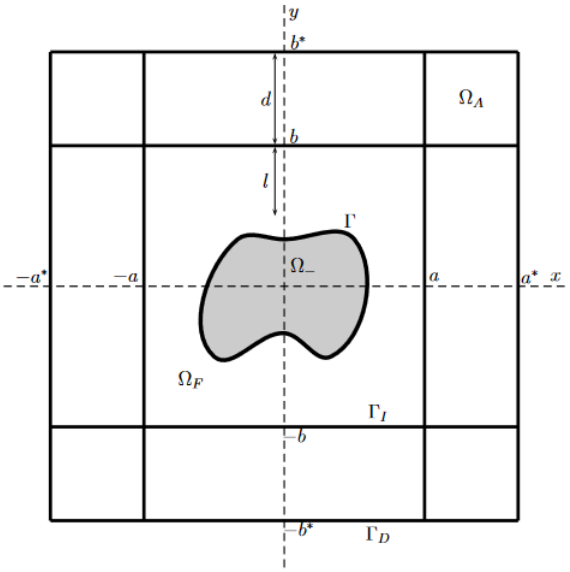


Figure 3.8 Domain of the PML problem taken from [53].

The 2D Cartesian PML problem is expressed as:

$$\begin{cases} \Delta u_F^s + k^2 u_F^s = 0 \quad \text{in } \Omega_F \\ \gamma_x^{-1}\partial_x\left(\gamma_x^{-1}\partial_x u_A^s\right) + \gamma_y^{-1}\partial_y\left(\gamma_y^{-1}\partial_y u_A^s\right) + k^2 u_A^s = 0 \text{ in } \Omega_A \\ \partial_\nu u_F^s = -\partial_\nu u^i \quad \text{on } \Gamma \\ u_F^s = u_A^s \text{ on } \Gamma_I \\ \partial_\nu u_F^s = -\partial_\nu u_A^s \quad \text{on } \Gamma_I \\ u_A^s = 0 \text{ on } \Gamma_D \end{cases} \tag{3.64}$$

where the functions in Ω_A are given by

$$\gamma_x(x) = \begin{cases} 1 & x < a \\ 1 + \frac{i}{k}\sigma_x(|x|) & a \leq |x| < a^* \end{cases}$$
$$\gamma_y(x) = \begin{cases} 1 & y < b \\ 1 + \frac{i}{k}\sigma_y(|y|) & b \leq |y| < b^* \end{cases} \tag{3.65}$$

and the ν is the normal vector oriented away from Ω_F. The σ_x and σ_y are absorbing functions given by

$$\sigma_x(x) = \sigma^*(x-a)^p$$
$$\sigma_y(y) = \sigma^*(y-b)^p \tag{3.66}$$

where σ^* is a constant and p is a positive integer.

NDDM can be naturally applied to the domain shown in Fig 3.8 to create 9 subdomains. In the usual way, the conditions on the interfaces between the subdomains must be rewritten as transmission conditions. Explicit conditions are provided on Γ_I and the implicit continuity conditions are imposed on artificial interfaces.

3.3.3 Numerical Tests for the PML

We apply the cross-point NDDM to Equation (3.64). The geometry in Figure 3.8 naturally splits into nine subdomains, with their corners treated as cross-points. We compare EMDA and Padé transmission operators on the artificial interfaces.

For the domain, let the scatterer be cylinder centered at the origin, meaning Γ is a circle of radius $r = 1$. Let the width of the PML shell be equal to the wavelength:

$$(b^* - b) = (a^* - a) = \lambda = \frac{2\pi}{k}.$$

The size of the physical domain will be fixed independent of the wavenumber. The physical domain containing the scatterer is a square of length 4: $a = b = 2$.

For the EMDA, let $\beta^{\chi} = 1/2k - \mathrm{i}k$ because it is heuristically the best for most problems. For the Padé operator, let rotation $\alpha = -\pi/4$, complexification $\varepsilon = -0.4k^{(1/3)}$, and order $N_p = 1, 2$. The Krylov subspace method ORTHODIR is used to solve the boundary iteration.

Table 3.9 reports iteration counts for the PML cross-point NDDM. We vary k over π, 2π, 3π, and 4π, and use meshes with $n_\lambda = 12$, 16, and 20 points per wavelength. The Padé operator is compared with scalar EMDA transmission.

Table 3.9 Iteration Number with respect to Different Operators and k

	EMDA			Padé 1			Padé 2		
k	12	16	20	12	16	20	12	16	20
π	139	164	184	76	93	103	74	77	81
2π	44	50	57	38	41	43	37	38	39
3π	37	40	43	36	37	40	34	36	37
4π	38	42	45	36	38	41	35	36	37

Increasing n_λ or k enlarges the linear systems and raises the cost per iteration. The average mesh spacing is $h = \frac{2\pi}{kn_\lambda}$. Nevertheless, iteration counts generally decrease in these experiments. This contrasts with the growth often observed for unpreconditioned Krylov solvers and illustrates the effectiveness of DDM preconditioning for the tested PML problems.

3.4 The Dual Transmission Operator

The usual NDDM algorithm uses the same transmission operator $\beta(k)$ on all artificial interfaces and for both sides of the interface, where k is the wavenumber of the incoming wave signal. Suppose the incident wave encounters a medium with multiple layers of varying indices of refraction n_i. The effective wave frequency in the ith layer

is kn_i. This is the **nonhomogeneous** scattering problem. When $n_i = 1$, for all layers, this is referred to as the homogeneous scattering.

In practice, the nonhomogeneous problem is more computationally challenging than the homogeneous problem. That is to say, the iteration number using $\beta(k)$ is typically larger when n_i are not all the same. The idea behind the dual transmission operator is to use a different transmission operator on each interface Σ_{ij}. Moreover, the dual transmission operator utilizes different operators $\beta_i(kn_j)$ and $\beta_j(kn_i)$ for each side of the interface. By incorporating more information about the refractive index, the convergence of the DDM can be accelerated. Refer to Subsection 2.3.2 for more details about the model and the implementation.

3.4.1 Motivation for the Dual Transmission Operator

Consider a Helmholtz transmission problem on two unbounded domains:

$$\begin{cases} \Delta u_1^{(m+1)} + k_1^2 u_1^{(m+1)} = 0, & x < 0 \\ \Delta u_2^{(m+1)} + k_2^2 u_2^{(m+1)} = 0, & x > 0 \\ (\partial_x + T_1) u_1^{(m+1)} = (\partial_x + T_1) u_2^{(m)}, & x = 0 \\ (-\partial_x + T_2) u_2^{(m+1)} = (-\partial_x + T_2) u_1^{(m)}, & x = 0. \end{cases} \tag{3.67}$$

Here $k_1 = kn_1$ and $k_2 = kn_2$, where k is the incident wavenumber and n_1, n_2 are the refractive indices. We assign transmission operators T_1 and T_2 to the two sides of the interface and, for simplicity, take them to be complex scalars. This model helps explain how distinct operators can improve convergence relative to $T_1 = T_2$.

The theoretical convergence rate of the problem in Equation (3.67) can be analyzed by utilizing a Fourier transform in the y variable. Let $\hat{u}_1(x, \xi)$ and $\hat{u}_2(x, \xi)$ be the Fourier transforms of u_1 and u_2, respectively. The transform of the PDEs can be expressed as ODEs in the x variable:

$$\partial_{xx} \hat{u}_i(x, \xi) - (\xi^2 - k_i^2) \hat{u}_i(x, \xi) = 0, \quad i = 1, 2. \tag{3.68}$$

Subject to the condition that $\hat{u}_1$ and $\hat{u}_2$ are continuous along the interface $x = 0$, it can be shown that solutions to the ODEs in Equation (3.68) are

$$\begin{cases} \hat{u}_i(x,\xi) = \hat{u}_i(0,\xi)e^{\mathrm{i}\lambda_i x}, & i = 1 \\ \hat{u}_i(x,\xi) = \hat{u}_i(0,\xi)e^{-\mathrm{i}\lambda_i x}, & i = 2, \end{cases} \tag{3.69}$$

where $\lambda_i = \sqrt{k_i^2 - \xi^2}$. We may substitute these solutions into the boundary conditions of the transmission problem in Equation (3.67) to obtain the two iteration formulas:

$$\begin{cases} \hat{u}_1^{(m+1)}(0,\xi) = \dfrac{-\mathrm{i}\lambda_2 + T_1}{\mathrm{i}\lambda_1 + T_1}\hat{u}_2^{(m)}(0,\xi) \\ \hat{u}_2^{(m+1)}(0,\xi) = \dfrac{-\mathrm{i}\lambda_1 + T_2}{\mathrm{i}\lambda_2 + T_2}\hat{u}_1^{(m)}(0,\xi). \end{cases} \tag{3.70}$$

The iteration formulas can be used to derive the overall iteration pattern of the transmission problem in Fourier space:

$$\hat{u}_i^{(m+2)}(0,\xi) = \frac{-\mathrm{i}\lambda_2 + T_1}{\mathrm{i}\lambda_1 + T_1}\frac{-\mathrm{i}\lambda_1 + T_2}{\mathrm{i}\lambda_2 + T_2}\hat{u}_i^{(m)}(0,\xi). \tag{3.71}$$

Hence, the convergence rate of the original problem in Equation (3.67) depends in some way on the spectral function

$$\rho(\xi) = \left|\frac{-\mathrm{i}\lambda_2 + T_1}{\mathrm{i}\lambda_1 + T_1}\frac{-\mathrm{i}\lambda_1 + T_2}{\mathrm{i}\lambda_2 + T_2}\right|. \tag{3.72}$$

for $\xi > 0$. We expect $\rho < 1$ for overall convergence of problem in Equation (3.67). The peak value $M = \max_{\xi>0}\rho(\xi)$ is a good indicator of the speed of the convergence. Smaller M corresponds to faster convergence in practice. To motivate the dual transmission operator, we will show that choosing $T_1 \neq T_2$ will result in lower M rather than choosing the same operators $T_1 = T_2$.

Equation (3.72) is the formula for a spectrum for the general case when T_1 and T_2 are different. The usual way of selecting the operators is to set $T_1 = T_2$. Two common examples are the Després operator $T = -\mathrm{i}k$ and the EMDA operator

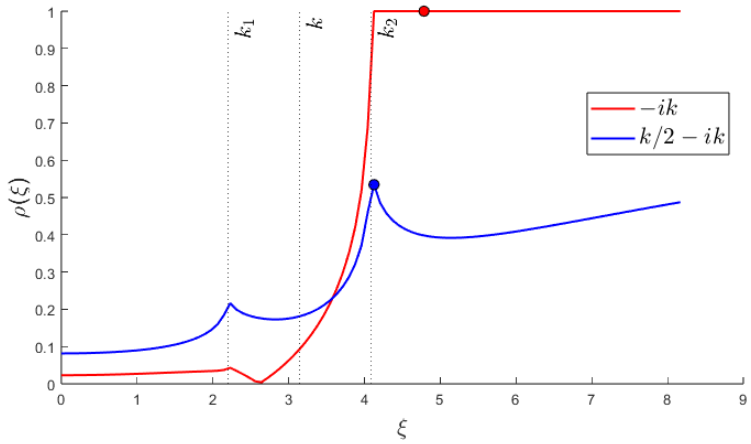


Figure 3.9 Spectrum of the Després and the EMDA operators with $n_1 = 0.7$ and $n_2 = 1.3$ are plotted. The maximum $M = \max_{\xi>0} \rho(\xi)$ is plotted as a filled circle.

$T = -\mathrm{i}k(1 + \mathrm{i}R)$. The spectrum for the Després and the EMDA with $R = 0.5$ are plotted in Figure 3.9 with $k = \pi$ $n_1 = 0.7$, $n_2 = 1.3$. The maximum M for the EMDA is nearly half of the max of the Després operator. In practice, the EMDA operator results in a much lower iteration number than the Després operator for general DDMs. This fact is reflected in the plot of the spectra.

Next, we compare the EMDA to dual transmission operators. There are several candidates for the dual transmission operators. We denote the indices by $i = 1, 2$ and $j = -i + 3$. We consider four choices for the dual scalar transmission operator:

(a) $T_i = kn_i - \mathrm{i}kn_i$,

(b) $T_i = kn_j - \mathrm{i}kn_j$,

(c) $T_i = k - \mathrm{i}kn_j$,

(d) $T_i = kn_j - \mathrm{i}k$.

The spectrum for these operators are plotted against the EMDA with $R = 0.5$ in Figure 3.10. We set $k = \pi$, $n_1 = 0.7$, and $n_2 = 1.3$.

Figure 3.10 shows similar spectral shapes for the tested operators. Their spectra peak near k_1 and k_2 and rise again for large ξ. The plotted range ends at $\xi_{\max} = 2k_2$. In descending order of the maximum value M, the operators are EMDA, (a), (d), (b),

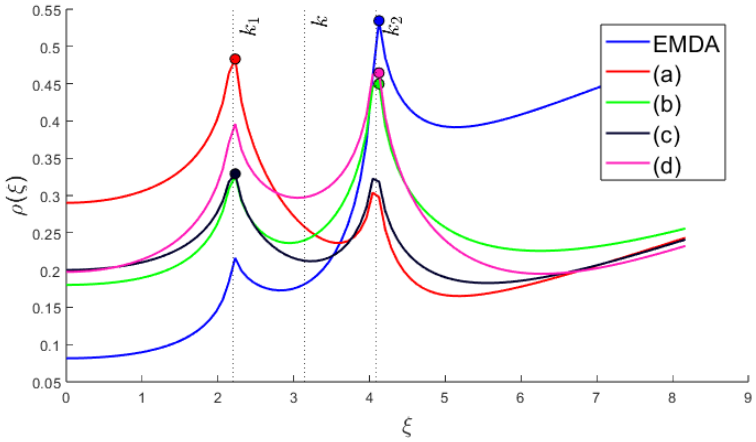


Figure 3.10 Plot shows the spectrum for the dual operators and their peak values.

and (c). The smaller maxima of the dual operators suggest faster convergence than standard EMDA.

Consider the following scattering problem on two subdomains:

$$\begin{cases} \dfrac{1}{\varepsilon_i}\Delta u_i^{(m+1)} + \dfrac{k^2 n_i^2}{\varepsilon_i} u_i^{(m+1)} = 0 & x \in \Omega_i \\ \varepsilon_i^{-1}\partial_{\mathbf{n_i}} u_i^{(m+1)} = g_i & x \in \Gamma_i \\ -\mathrm{i}k u_i^{(m+1)} + \varepsilon_i^{-1}\partial_{\mathbf{n}_i} u_i^{(m+1)} = g_{i0} & x \in \Sigma_{i0} \\ T_i u_i^{(m+1)} + \varepsilon_i^{-1}\partial_{\mathbf{n_i}} u_i^{(m+1)} = g_{ij}^{(m)} & x \in \Sigma_{ij}, \end{cases} \tag{3.73}$$

where the domain is an annulus divided in half along the line $y = 0$. The inner radius is $r_1 = 1$, and the outer is $r_2 = 3$. The incident wave is $u^{\text{inc}} = e^{-\mathrm{i}kx}$, where $k = \pi$. We choose the permittivity in each subdomain to be $\epsilon_i = 1$, and let $n_1 = 0.7$, $n_2 = 1.3$. A sketch of the domain is shown in Figure 3.11. A plot of the numerical solution is shown in Figure 3.12. The problem is symmetric when $n_1 = n_2 = 1$ which is what is shown on the left. The plot with $n_1 = 1.3$ and $n_2 = 0.7$ is shown on the right.

We test the four dual scalar operators (a)–(d) on Equation (3.73). Jacobi and ORTHODIR converge for all operators. Table 3.10 shows that the dual operators require fewer ORTHODIR iterations than EMDA with $R = 1/2$. The dual operators have equal Jacobi iteration counts, approximately half the EMDA count.

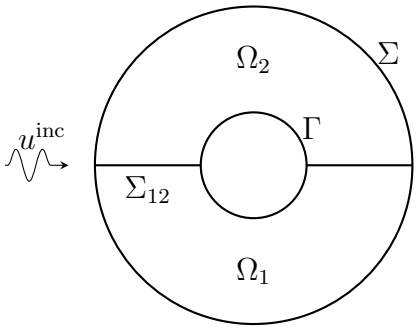


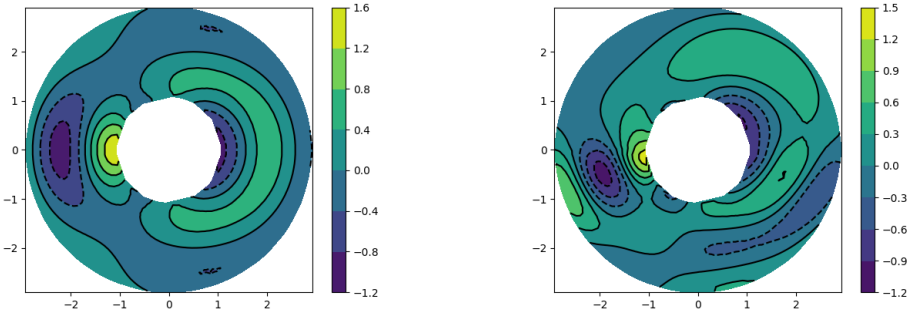
Figure 3.11 Geometry to test the dual transmission EMDA operators.

Table 3.10 Iteration Numbers for the Scalar Dual Operators

	EMDA $k/2 - \mathrm{i}k$	(a) $k_i - \mathrm{i}k_i$	(b) $k_j - \mathrm{i}k_j$	(c) $k - \mathrm{i}k_j$	(d) $k_j - \mathrm{i}k$
Jacobi	159	81	81	81	81
ORTHODIR	32	25	26	25	26

In this example, dual EMDA operators outperform standard EMDA for the nonhomogeneous NDDM problem. The numerical behavior agrees with the Fourier analysis of the unbounded transmission problem. This motivates dual extensions of other wavenumber-dependent operators $\beta(k)$.

The boundary IE operator $\mathcal{T}_\epsilon^S u := 2D_{k+\mathrm{i}\epsilon}u$ discussed in Section 3.2 depends on the wavenumber. This operator can also be generalized with dual transmission technique. On the interface Σ_{ij}, we simply let $T_i = 2D_{k_j+\mathrm{i}\epsilon}$, where k_j is the effective



(a) Homogeneous solution **(b)** Nonhomogeneous solution

Figure 3.12 Numerical solutions for the scattering problem with and without homogeneity.

wavenumber in subdomain Ω_j. The Padé-localized square root operator discussed in Section 3.1 also depends on k, but it is not as simply generalized. The next section discusses how the PLS operator can be modified to be used as a dual transmission operator.

3.4.2 Dual Transmission Padé Operator

Let u_i be the solution on subdomain Ω_i which has constant index of refraction n_i. Let us also assume the complexification parameter ε to be constant. For domain Ω_i, the Padé dual transmission operator on the interface Σ_{ij} is defined as

$$T_i u_i = -\mathrm{i}kn_j \left(C_0 + \frac{1}{(kn_j + \mathrm{i}\varepsilon)^2} \sum_{\ell=1}^{N_p} A_\ell \Delta_S \left(1 + \frac{B_\ell}{(kn_j + \mathrm{i}\varepsilon)^2} \Delta_S \right)^{-1} \right) u_i, \tag{3.74}$$

where Δ_S is the Laplace-Beltrami operator on the curve S. We let the auxiliary unknown for each $\ell = 1, \ldots, N_p$ be defined as:

$$\left(1 + \frac{B_\ell}{(kn_j + \mathrm{i}\varepsilon)^2} \Delta_S \right) \varphi_{i,\ell} = u_i. \tag{3.75}$$

The unknowns $\varphi_{i,\ell}$ are for domain Ω_i and they depend on kn_j. These auxiliary unknowns do not provide all the information needed for the dual transmission update. The sum of the operators is

$$\begin{aligned}(T_i + T_j)u_j = -\mathrm{i}k(n_i + n_j)C_0 u_j &- \frac{\mathrm{i}kn_i}{(kn_i + \mathrm{i}\varepsilon)^2} \sum_{\ell=1}^{N_p} A_\ell \Delta_S \varphi_{j,\ell} \\ &- \frac{\mathrm{i}kn_j}{(kn_j + \mathrm{i}\varepsilon)^2} \sum_{\ell=1}^{N_p} A_\ell \Delta_S \left(1 + \frac{B_\ell}{(kn_j + \mathrm{i}\varepsilon)^2} \Delta_S \right)^{-1} u_j \end{aligned} \tag{3.76}$$

The sum contains an inverse operator, motivating an additional auxiliary unknown ψ satisfying

$$\left(1 + \frac{B_\ell}{(kn_j + \mathrm{i}\varepsilon)^2} \Delta_S \right) \psi_{j,\ell} = u_j. \tag{3.77}$$

Hence, the variational form of the update is given by

$$\begin{aligned}\int_{\Sigma_{ij}} g_{ij}^{(n+1)} v ds &= -\int_{\Sigma_{ij}} g_{ji}^{(n)} v ds + \int_{\Sigma_{ij}} (T_i + T_j) u_j^{(n+1)} v ds \\ &= -\int_{\Sigma_{ij}} g_{ji}^{(n)} v ds - \mathrm{i}k(n_j + n_i) C_0 \int_{\Sigma_{ij}} u_j^{(n+1)} v \, ds + \\ &\quad \frac{\mathrm{i}kn_j}{(kn_j + \mathrm{i}\varepsilon)^2} \sum_{\ell=1}^{N_p} A_\ell \int_{\Sigma_{ij}} \nabla \psi_{j,\ell}^{(n+1)} \cdot \nabla v ds + \\ &\quad \frac{\mathrm{i}kn_i}{(kn_i + \mathrm{i}\varepsilon)^2} \sum_{\ell=1}^{N_p} A_\ell \int_{\Sigma_{ij}} \nabla \varphi_{j,\ell}^{(n+1)} \cdot \nabla v ds.\end{aligned} \tag{3.78}$$

At the discrete level, the φ unknowns are naturally computed as a byproduct of the Schur complement procedure. The ψ, on the other hand, requires an additional system to be constructed and then solved. The FEM system for $\psi_{j,\ell}^{(n+1)}$ comes from using the Galerkin projection method on the bilinear equation $a(\psi, v) = L(v)$, where

$$\begin{aligned} a(\psi, v) &= \int_{\Sigma_{ij}} \psi v - \frac{B_\ell}{(kn_j + \mathrm{i}\varepsilon)^2} \nabla \psi \cdot \nabla v ds \\ L(v) &= \int_{\Sigma_{ij}} u_j^{(n+1)} v ds \end{aligned} \tag{3.79}$$

The left-hand side assembly for $\psi_{j,\ell}$ is actually naturally computed as part of the usual process for implementing Padé with nodal finite element method. In the implementation, a submatrix of the R_i can be extracted for efficiency.

3.4.3 Numerical Results for the Dual Transmission Operators

We now test dual transmission operators on a nonhomogeneous scattering problem. A cylindrical scatterer with boundary Γ is illuminated by a plane wave from the left and surrounded by two concentric layers with refractive indices n_1 and n_2. An absorbing condition is imposed on the outer surface Σ. Introducing interfaces along $y = 0$ and $x = 0$ divides the geometry into eight subdomains; see Figure 3.13. The local governing equations are those of Equation (3.73), with dual transmission operators T_i.

We set $r_1 = 1$, $r_2 = 3$, and $k = \pi$, and use 18 points per wavelength. The cross-point algorithm is tested with EMDA, Padé, and BIE operators. We use $\chi = 0.5$ for EMDA, $N_p = 4$ for Padé, and $\varepsilon = 0.5$ for BIE. ORTHODIR solves the interface system.

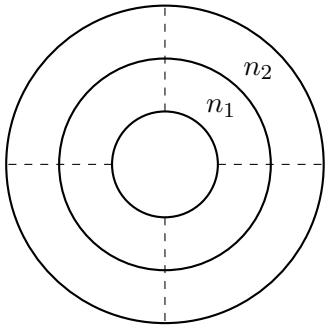


Figure 3.13 Two-layer nonhomogeneous scattering geometry. The material layers and the dashed artificial interfaces divide the computational domain into eight subdomains and introduce cross-points.

The results for the DDM iteration numbers are shown in Table 3.11. The DDM was tested using both the regular version of the transmission operator which depends on the wavenumber k, and the generalized dual operator which incorporates k_i and k_j. The results show that the dual version of each operator results in faster convergence compared to the regular version. The only exception is $n_1 = 1.3$,$n_2 = 0.7$ where the Padé operator performed the same with and without the dual. The results show that dual transmission operator is effective at utilizing additional information about the nonhomogeneous problem to improve upon the regular operator.

Table 3.11 Comparison of Iteration Numbers with and without the Dual Transmission Operator

	EMDA		Padé		BIE	
	Regular	Dual	Regular	Dual	Regular	Dual
$n_1 = 0.7, n_2 = 1.3$	55	49	38	35	28	24
$n_1 = 1.3, n_2 = 0.7$	45	43	26	26	19	17

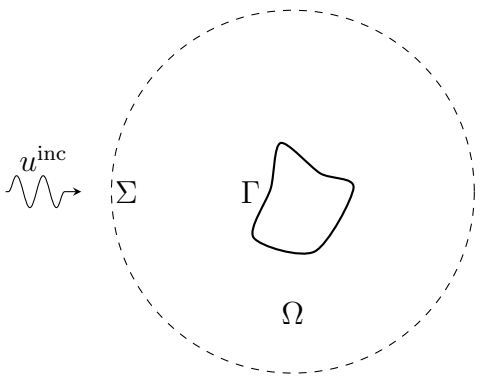


Figure 3.14 Sketch of a scattering geometry with Γ being the scatterer and Σ being the truncating surface.

3.5 Adaptive Radiation Condition

This section introduces the adaptive radiation condition (ARC) as another application of the cross-point algorithm. Approaches to unbounded scattering domains include integral equations, infinite elements, and domain truncation. ARC is a truncation method whose boundary data are updated iteratively to recover the exact radiation condition [12].

An absorbing boundary condition (ABC) makes the computational domain finite by imposing a local approximation on the truncation surface. This introduces error. A higher-order ABC or a more distant boundary can reduce that error, but increases computational cost. ARC instead updates the boundary data so that, at convergence, the truncated continuous problem recovers the exact radiation condition.

Let Γ be the surface of a scatter on which an impedance condition is prescribed. Let the domain of the scattering problem Ω be truncated at a surface Σ which encloses the scatterer. Let C be any sufficiently smooth closed contour contained in $\overline{\Omega}$. Consider the problem

$$\begin{cases} \Delta u + k^2 u = 0 & x \in \Omega \\ (\partial_{\mathbf{n}} + \beta) u = -(\partial_{\mathbf{n}} + \beta_1) u^{\text{inc}} & x \in \Gamma \\ (\partial_{\mathbf{n}} - \mathrm{i}k) u = g_a & x \in \Sigma \end{cases} \tag{3.80}$$

where the adaptive boundary data g_a are defined by

$$g_a(w) := -\mathrm{i}kw + \partial_{\mathbf{n}_C} w, \tag{3.81}$$

and w is given by

$$w(u) := -V(\partial_{\mathbf{n}_C} u|_C) - N(u|_C). \tag{3.82}$$

The vector $\mathbf{n}_C$ is the unit normal pointing outward of the curve C. The V and N are the single- and double-layer potentials defined by

$$Vp(x) = \int_C G(x,y)p(y)ds_y \tag{3.83}$$

$$N\lambda(x) = -\int_C \partial_{\mathbf{n}_y} G(x,y)\lambda(y)ds_y \tag{3.84}$$

The function $G(x,y)$, for $x \neq y$, is the Green's Kernel related to the outgoing solution of the 2D Helmholtz equation.

The significance of the problem in Equation (3.80) is that it is equivalent to the exterior scattering problem in Equation (A.1) [12]. See Appendix A. The problem in Equation (3.80) is bounded, yet its solution is equivalent to that of an unbounded problem. The boundary data enforced on Σ is fundamental to the ARC method.

Now, we introduce the ARC problem. The right-hand side g_a depends on the solution u. The main idea behind the ARC is to solve Equation (3.80) iteratively. The exterior boundary data $g_a^{(0)}$ is initialized with zero. Then the BVP can be solved and used to compute the next boundary data $g_a^{(n)} = g_a(w(u^{(n)}))$. The iterative ARC scheme is

$$\begin{cases} \Delta u^{(n)} + k^2 u^{(n)} = 0 & x \in \Omega \\ (\partial_{\mathbf{n}} + \beta)u^{(n)} = -(\partial_{\mathbf{n}} + \beta)u^{\mathrm{inc}} & x \in \Gamma \\ (\partial_{\mathbf{n}} - \mathrm{i}k)u^{(n)} = g_a^{(n-1)} & x \in \Sigma \end{cases} \tag{3.85}$$

with the update given by

$$\begin{aligned} g_a^{(n)} &= -\mathrm{i}kw^{(n)} + \partial_{\mathbf{n}_C} w^{(n)}, \\ w^{(n)} &= -V(\partial_{\mathbf{n}_C} u^{(n)}|_C) - N(u^{(n)}|_C). \end{aligned} \tag{3.86}$$

The curve C is a closed contour contained in $\overline{\Omega}$. The bounded PDE problem can be discretized with FEM, while boundary integral evaluation supplies the update. ARC combines naturally with NDDM: both are iterative, and NDDM supplies interface normal derivatives through its Robin data. After decomposing Ω into N_d subdomains, the update of $g_a^{(n)}$ is performed after the local solves. If C is formed from artificial interfaces, the Neumann data needed for ARC are available from the NDDM iteration. Figure 3.15 illustrates this choice, with C shown as four bold dashed interfaces. The combination also permits parallel local solves.

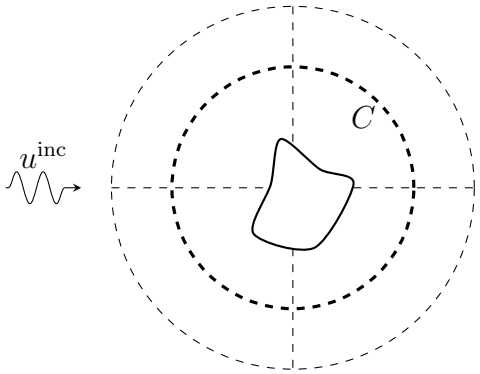


Figure 3.15 Sketch of the NDDM-ARC domain. The collection of artificial interfaces in bold dashed lines can be used for the ARC.

Let us write the equations for the general NDDM ARC. The domain Ω is split into non-overlapping domains Ω_i for $i = 1$ to N_d. We denote the restrictions of Γ, Σ, and u to the local domain $\overline{\Omega}_i$ by Γ_i, Σ_i, and u_i. On the artificial interface $\Sigma_{ij} = \overline{\Omega}_i \cap \overline{\Omega}_j$, we enforce the Robin condition with a transmission operator β_s. The NDDM ARC is expressed as

$$\begin{cases} (\Delta + k^2)u_i^{(n)} = 0 & x \in \Omega_i \\ (\partial_{\mathbf{n}_i} + \beta)u_i^{(n)} = g^{\text{inc}} := -(\beta + \partial_{\mathbf{n}})u^{\text{inc}} & x \in \Gamma_i \\ (\partial_{\mathbf{n}_i} + T)u_i^{(n)} = g_{ij}^{(n-1)} & x \in \Sigma_{ij} : j \in \Lambda_i \\ (\partial_{\mathbf{n}_i} - \mathrm{i}k)u_i^{(n)} = g_a^{(n-1)} & x \in \Sigma_i \end{cases}, \quad \forall i. \tag{3.87}$$

On an interface Σ_{ij} belonging to C, both $u_i^{(n)}$ and $u_j^{(n)}$ are available. We choose the trace from the side whose outward normal agrees with the orientation of C. Let Λ_C index the subdomains used in the ARC update. The update formulas are

$$g_{ij}^{(n)} = -g_{ji}^{(n-1)} + 2Tu_j^{(n)} \tag{3.88}$$

$$\lambda^{(n)} := \sum_{i \in \Lambda_C} u_i^{(n)}\Big|_C \tag{3.89}$$

$$p^{(n)} := \sum_{i \in \Lambda_C} \left(g_{ji}^{(n-1)} - Tu_i^{(n)}\right)\Big|_C \tag{3.90}$$

$$g_a^{(n)} = (-\mathrm{i}k + \partial_{\mathbf{n}})\left(-V(\lambda^{(n)}) - N(p^{(n)})\right). \tag{3.91}$$

The weak form of Equation (3.87) can be derived in the usual way and is given by

$$\int_{\Omega_i} (\nabla u_i^{(n)} \cdot \nabla v_i - k^2 u_i^{(n)} v_i)\mathrm{d}x + \int_{\Gamma_i} \beta u_i^{(n)}\, v_i \,\mathrm{d}s + \int_{\Sigma_{ij}} Tu_i^{(n)} v_i \mathrm{d}s = \int_{\Gamma_i} g^{\text{inc}}\, v_i \,\mathrm{d}s + \int_{\Sigma_i} g_a^{(n-1)} v_i \mathrm{d}s + \int_{\Sigma_{ij}} g_{ij}^{(n-1)} v_i \mathrm{d}s. \tag{3.92}$$

The weak form of the update formula is also required. When we test it with a function $v \in H^1(\Sigma)$ and integrate over the outer surface Σ we get:

$$\begin{aligned} \int_\Sigma g_a^{(n)} v \,\mathrm{d}s &= \int_\Sigma (-\mathrm{i}k + \partial_{\mathbf{n}})\left(-V(\lambda^{(n)}) - N(p^{(n)})\right) v \,\mathrm{d}s \\ &= \int_\Sigma (-\mathrm{i}k + \partial_{\mathbf{n}})\left(-V(\lambda^{(n)})\right) v \,\mathrm{d}s + \int_\Sigma (-\mathrm{i}k + \partial_{\mathbf{n}})\left(-N(p^{(n)})\right) v \,\mathrm{d}s \\ &= \int_\Sigma (\mathrm{i}kV + N^T)\left(\lambda^{(n)}\right) v \,\mathrm{d}s + \int_\Sigma (\mathrm{i}kN - D)\left(p^{(n)}\right) v \,\mathrm{d}s. \end{aligned} \tag{3.93}$$

The notation D is the hypersingular integral operator and the notation N^T stands for the adjoint operator:

$$\int_\Sigma v N^T \left(\partial_{\mathbf{n}_C} u^{(n)}|_C\right) \, \mathrm{d}s = \int_C \partial_{\mathbf{n}_C} u^{(n)} N(v) \, \mathrm{d}s. \tag{3.94}$$

The weak formulations of the PDE and update define the continuous NDDM-ARC method. The FEM and boundary integral matrices need to be assembled only once before iteration.

One can choose $C = \Gamma$, the surface of the scatterer. Out of the three choices Γ is the shortest in length. Thus it leads to the smallest discrete system. Also the normal derivative $\partial_{\mathbf{n}_C} u^{(n)}$ on Γ is supplied by the incident wave. It only needs to be computed once. We would have

$$\partial_{\mathbf{n}_C} u^{(n)} = g^{\mathrm{inc}} - \beta \sum_{i \in \Lambda_C} u_i^{(n)} \Big|_C . \tag{3.95}$$

The g^{inc} is given. Hence, the bilinear form in the update becomes:

$$\begin{aligned} \int_\Sigma g_a^{(n)} v \, \mathrm{d}s &= \int_\Sigma (\mathrm{i}kV + N^T) \left(\partial_{\mathbf{n}_C} u^{(n)}|_C\right) v \, \mathrm{d}s + \int_\Sigma (\mathrm{i}kN - D) \left(u^{(n)}|_C\right) v \, \mathrm{d}s. \\ &= \int_\Sigma (\mathrm{i}kV + N^T) \left(g^{\mathrm{inc}}\right) v \, \mathrm{d}s + \cdots \\ &\int_\Sigma (-\mathrm{i}k\beta V - \beta N^T + \mathrm{i}kN - D) \left(u^{(n)}|_C\right) v \, \mathrm{d}s. \end{aligned} \tag{3.96}$$

The integral term on the left needs to be assembled once and the update integral on the right does not need normal derivative on corner nodes.

Another alternative is to choose $C = \Sigma$, the outer truncation surface on which the ARC condition is imposed. Doing so will make the bilinear form for the update equation symmetric which may be desirable. The disadvantage of this method is that the Neumann data on Σ is not by-product of the NDDM. An extra storage of the previous Robin data would be required on the outer surface.

3.5.1 Numerical Results

We test the cross-point ARC algorithm with nonscalar transmission operators, choosing C as a closed contour formed by artificial interfaces. A sound-hard cylindrical scatterer of radius 1 is illuminated by an incident plane wave u^{inc} from the left. The domain is truncated at the circle Σ, $|x| = 3$, and divided into eight subdomains; see Figure 3.16. The NDDM system is expressed in Equation (3.87).

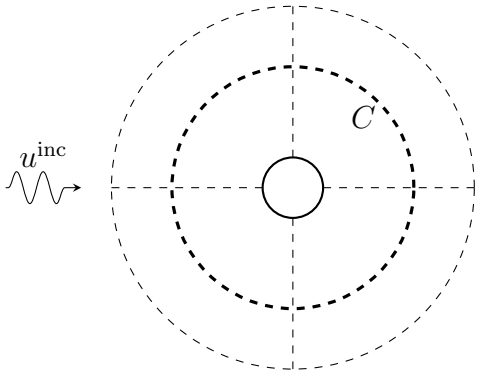
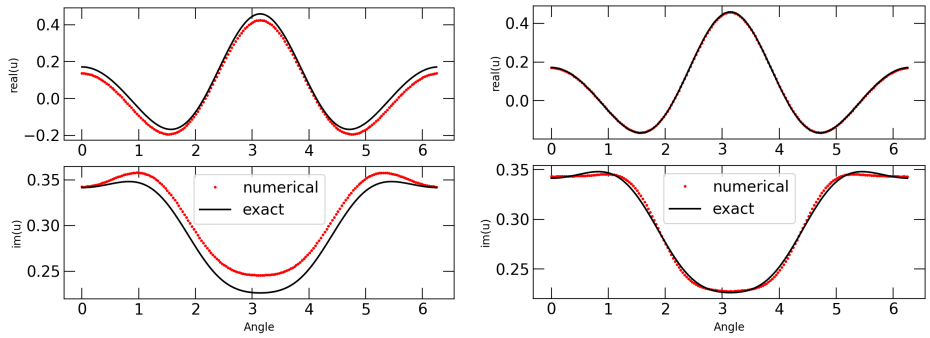


Figure 3.16 Sketch of the NDDM-ARC domain for the numerical tests.

We will compare the results for scalar transmission operator and the Padé operator.

First, we will demonstrate the quantitative difference between the ARC and the ABC. One of the drawbacks of the ABC is that the truncation surface must be placed sufficiently far from the scatterer with respect to the wavelength of the incident wave. Otherwise, the accuracy of the numerical solution will contain a relatively large error. An example comparing the numerical solution between the ABC and ARC is shown in Figure 3.17. The simulation was performed on an annulus with inner radius $r_1 = 1$ and outer radius $r_2 = 3$. The wavenumber was chosen to be $k = \pi/2$ which is relatively low compared to the domain size. The error on the exterior circle is comparatively much larger for the ABC than the ARC.

A test was performed for $k = \pi$ comparing the convergence of the cross-point algorithm with the EMDA operator and the Padé operator. The results are shown in



(a) ABC **(b)** ARC

Figure 3.17 A comparison of the error between the ABC and the ARC.

Table 3.12. The experiment demonstrates that the rate of convergence of the DDM can be accelerated with the use of the Padé operator.

Table 3.12 Iteration Numbers for the ARC with $k = \pi$

	Jacobi	ORTHODIR
EMDA	109	44
$N_p = 1$	81	36
$N_p = 2$	74	31
$N_p = 3$	73	29
$N_p = 4$	70	28

3.6 Acceleration of the Convergence of the NDDM Algorithm

We extend the method of Subsection 2.4.3 to accelerate convergence. The extension enforces global continuity at additional interface nodes near each cross-point.

Introduce an integer $\delta \geq 0$. Along each interface emanating from a cross-point, the nearest δ nodes are assigned to the globally continuous corner system. The original method corresponds to $\delta = 0$. Implementation requires changing the degree-of-freedom classification of these nodes.

We start with a conforming mesh of the non-overlapping domains. We distinguish each node of the mesh as one of the three: independent, interface, or corner. See Figure 3.1. The three types of nodes are colored in red, blue, and black;

respectively. The distinction is important in the assignment of the degree of freedom for each node as is described in Subsection 2.4.3. The transmission operators on the artificial interfaces Σ_{ij} are the ones used previously, EMDA, Pade, BIE [15, 17, 62].

3.6.1 Numerical Results

We test how increasing δ affects DDM convergence and how the effect varies with mesh size and wavenumber. The implementation accepts δ as an additional parameter.

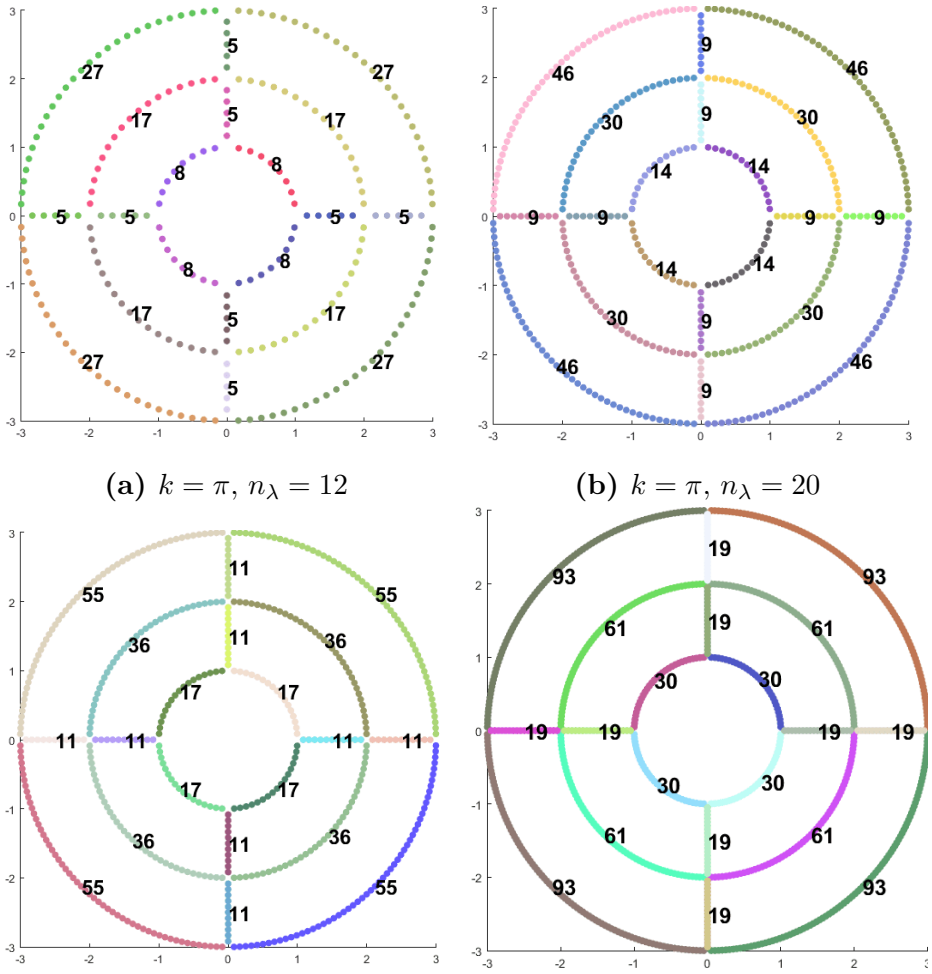
We choose our computational domain to be the annulus since the analytical solution is known for this particular geometry. We chose an 8-domain dartboard-style decomposition since it naturally introduces several cross-points. We chose an annulus with an inner radius $r_1 = 1$ and outer radius $r_2 = 3$. We varied the wavenumber $k = \pi, 2\pi$ and the points per wavelength $n_\lambda = 12, 20$. The mesh size can be computed with $h = \frac{2\pi}{kn_\lambda}$. Figure 3.18 shows the four cases.

The number of nodes along the interfaces varies for each case. In the case (a), $k = \pi, n_\lambda = 12$, the shortest boundaries only have five nodes. Therefore, $\delta > 2$ will change the short boundaries entirely into cross-points. Table 3.13 shows the results of increasing δ for the four test cases.

Table 3.13 reports the EMDA iteration counts needed to reduce the initial residual by a factor of 10^{-6} as δ varies. In all four cases, increasing δ improves convergence.

The first observation is simple. In all four cases, the iteration number is monotonically (not strictly) decreasing with respect to δ. NDDM can be thought of as an iteration of Robin data on the artificial interfaces. The size of the Robin data is equal to two times the total number of interface nodes.

The size of the improvement varies across the four cases. Recall that δ counts the nodes near each corner that are assigned global continuity. Cases (a)–(d) have progressively more mesh nodes, so a fixed δ modifies a smaller proportion of each



(a) $k = \pi$, $n_\lambda = 12$ **(b)** $k = \pi$, $n_\lambda = 20$

(c) $k = 2\pi$, $n_\lambda = 12$ **(d)** $k = 2\pi$, $n_\lambda = 20$

Figure 3.18 Quantity of nodes on each boundary are shown in all test cases

interface. Table 3.13 suggests that iteration reduction is related to the physical extension length $h\delta$.

3.6.2 Comparison with other Transmission Operators

We repeat the EMDA experiments with Padé and complexified IE transmission operators [15, 62]. Figure 3.19 plots the iteration counts.

The comparison provides further insight into the continuity extension. Iteration counts need not decrease monotonically with δ. The decrease is approximately steady up to a value δ_0, after which it becomes more pronounced, particularly in case (d).

Table 3.13 Iteration numbers with respect to δ

(a)

δ	Iter
0	36
1	34
2	31
3	28
4	27
5	22
6	17
7	12

(b)

δ	Iter
0	44
1	43
2	41
3	39
4	37
5	36
6	35
7	33
8	31
9	29
10	27
11	25
12	21
13	17
14	10

(c)

δ	Iter
0	40
2	38
4	35
6	31
8	30
10	29
12	27
14	23
16	14

(d)

δ	Iter
0	47
2	46
4	45
6	43
8	40
10	38
12	37
14	36
16	35
18	34
20	33
22	32
24	29
26	24
28	17

For large δ, the counts for the three operators approach one another. In this regime, an inexpensive transmission operator may be preferable because the iteration counts are similar.

Next, we fix $\delta = 3$ and vary the mesh size to study convergence as the discrete system grows. The annulus remains divided into eight subdomains. We use $k = \pi, 2\pi$ and $n_\lambda = 12$, 16, 20, comparing EMDA, Padé with $N_p = 2$, and the IE operator with $\epsilon = 0.5$.

Table 3.14 Iteration Numbers for Fixed δ

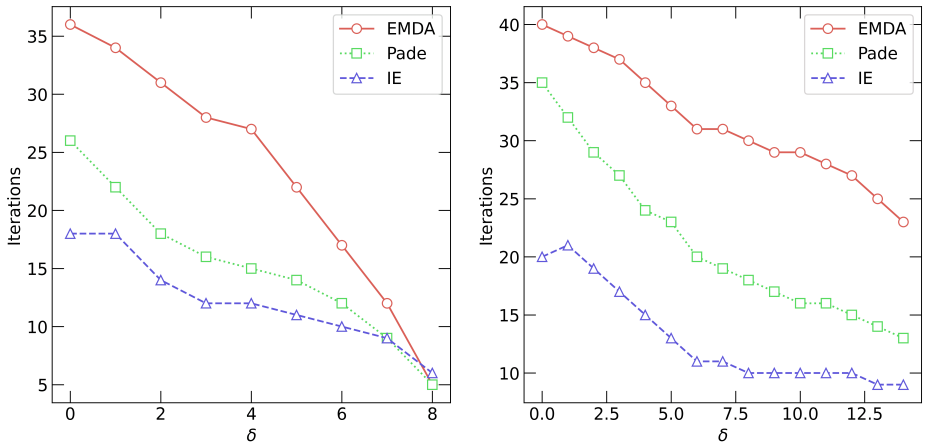
n_λ	EMDA	Padé	IE
12	28	15	12
16	35	20	14
20	39	24	14

(a) $k = \pi$

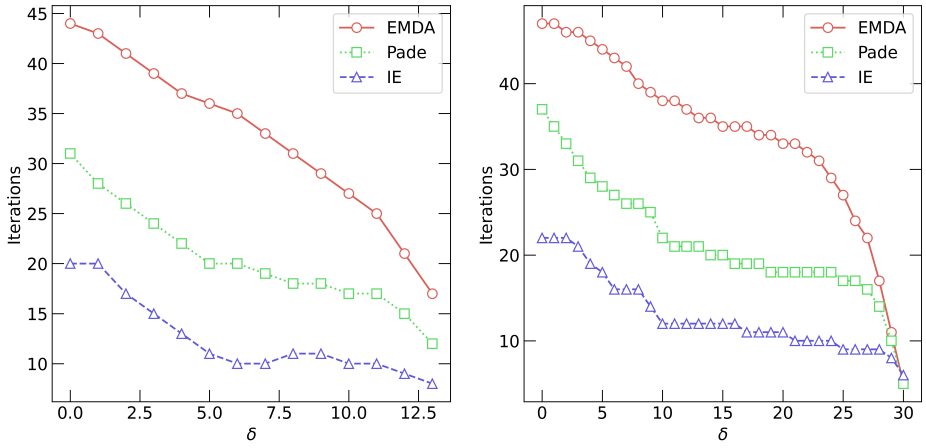
n_λ	EMDA	Padé	IE
12	36	26	15
16	41	28	17
20	45	30	19

(b) $k = 2\pi$

Table 3.14 reports the Krylov iteration counts needed to reduce the relative residual below 10^{-6}.



(a) $k = \pi$, $n_\lambda = 12$ **(b)** $k = \pi$, $n_\lambda = 20$



(c) $k = 2\pi$, $n_\lambda = 12$ **(d)** $k = 2\pi$, $n_\lambda = 20$

Figure 3.19 Plots indicate the iteration number for the four test cases with respect to δ and for the three transmission operators.

3.6.3 Partially Coated Scatterer

We now consider a partially coated scatterer whose surface contains reflective and transmitting regions. Waves can enter its interior through the transmitting regions. This example tests the method on a more challenging geometry.

We have meshed an annulus with mixed boundary conditions for the partially coated scatterer simulation. The scattering surface Γ is now composed of sub-regions Γ_N with a reflection condition and sub-regions Γ_T with the continuity condition. See Figure 3.20. Let u be the wave inside the region Ω and let u_s be the wave inside the scatterer Ω_s. We model the boundary conditions for this problem as:

$$\partial_n u = f \quad \text{on} \quad \Gamma_N \tag{3.97}$$

and

$$\begin{cases} u = u_s \\ \partial_n u = \partial_n u_s \end{cases} \quad \text{on} \quad \Gamma_T \tag{3.98}$$

The conditions in Equation (3.98) are rewritten as Robin conditions with the EMDA transmission operator in the usual way.

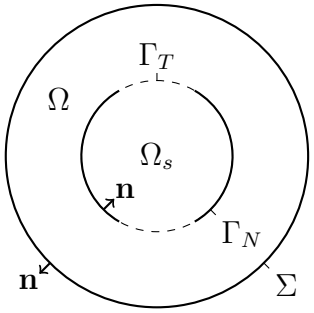
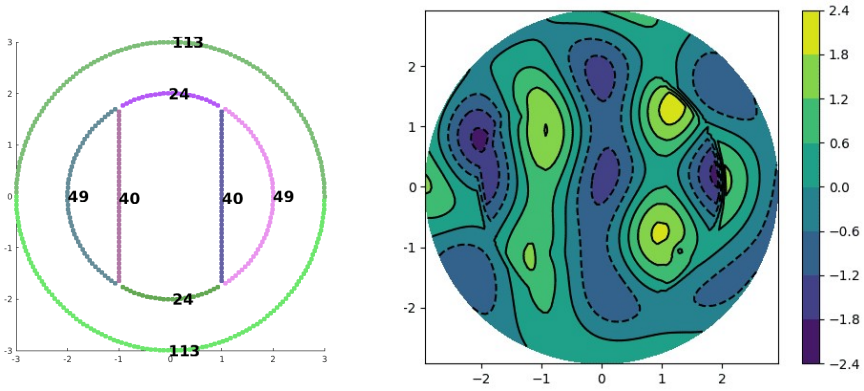


Figure 3.20 Geometry for partially coated scatterer.

For this problem, the region inside the scatterer is a computational domain because the wave is able to penetrate into Ω_s through the surface Γ_T. We split the interior of the scatterer into 3 sub-domains, but keep outer domain Ω whole. There are 4 cross-points on this mesh and they lie on the scatterer at the points where Γ_N meets Γ_T.

We fix $k = \pi$ and choose $r_1 = 2, r_2 = 3$ to balance the subdomain sizes, with mesh size $h = 1/12$. Figure 3.21 shows the interface nodes and numerical solution. Figure 3.22 reports iteration counts as δ varies. The operator parameters match the previous comparison except that the IE complexification parameter is $\varepsilon = 0.9$, which gave a slightly lower count in this test. IE converges fastest and EMDA slowest. Extending continuity farther from the corners reduces the iteration count.



(a) Points on the interface. **(b)** Numerical simulation.

Figure 3.21 The discretization of the interfaces are shown on the left and the real part of the numerical solution on the right.

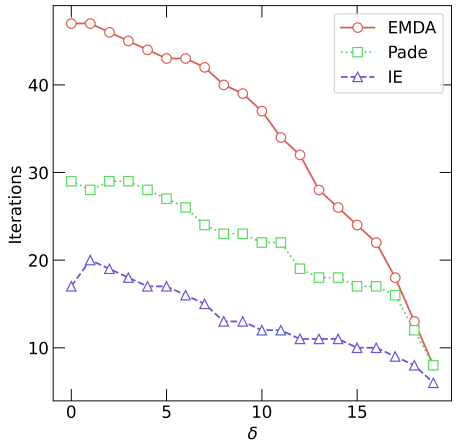


Figure 3.22 Iteration numbers for the NDDMs on the partially coated scatterer with respect to δ and for the different transmission operators.

3.7 Concluding Remarks

Domain decomposition for the Helmholtz equation builds on an established literature for linear elliptic problems. The non-coercive Helmholtz operator motivates Robin interface conditions, which support well-posed local problems and make NDDM a natural approach.

Two ideas guide the methods in this chapter. First, the transmission operator strongly influences NDDM convergence. Constructing an operator that performs well across many problems is difficult. A useful strategy is to study a simplified

transmission problem with a known or approximable DtN map and use that approximation to design the interface operator.

Second, non-overlapping nodal FEM decompositions require a consistent treatment of cross-points. Desirable properties include stability, convergence, and limited duplication of interface unknowns. The continuous cross-point algorithm of Bendali and Boubendir provides these features for scalar transmission operators. Our numerical experiments also demonstrate convergence with the nonscalar operators considered here.

Replacing scalar transmission by more advanced operators reduces iteration counts by up to a factor of five in the reported tests. The PML and ARC applications illustrate the flexibility of the cross-point framework. These ideas also motivate extensions to other linear elliptic problems for which the framework is applicable.

A remaining theoretical task is to prove convergence of the cross-point algorithm for general transmission operators. The original argument uses symmetry of the bilinear operator, which is unavailable for general nonscalar choices. The numerical evidence motivates a convergence analysis that does not rely on that symmetry.

CHAPTER 4

BACKGROUND FOR THE MONGE-AMPÈRE EQUATION

The Monge-Ampère equation in R^n is $\det(D^2u) = f(x, u, Du)$, with u convex. It is semilinear in one dimension (1D) and fully nonlinear in two dimensions (2D) or higher. This chapter reviews weak well-posedness for the Dirichlet problem, convergent discretizations for viscosity solutions, and nonlinear solvers that can be combined with domain decomposition.

4.1 Viscosity Solution

This section reviews viscosity solutions for nonlinear second-order PDEs. Most definitions and theorems follow Crandall et al. [29]. The framework applies to equations of the form

$$F(x, u(x), \nabla u(x), D^2u(x)) = 0, \quad x \in \overline{\Omega} \tag{4.1}$$

where F is real-valued and Ω is a bounded open set in $\mathbb{R}^n$ and $\overline{\Omega}$ is its closure. The notation $D^2u(x)$ represents the $n \times n$ **Hessian matrix** whose elements are given by

$$(D^2u)_{i,j} = \frac{\partial^2 u}{\partial x_i \partial x_j}. \tag{4.2}$$

When the PDE in Equation (4.1) is equipped with the condition

$$F(x, r, p, X) \leq F(x, s, p, Y), \quad \text{whenever } r \leq s \text{ and } Y \leq X, \tag{4.3}$$

the PDE is said to be a **proper elliptic equation**. The condition $Y \leq X$ means the matrix $X - Y$ is positive semidefinite. There is a more general class of PDEs called degenerate elliptic equations. The condition "whenever $r \leq s$" can be relaxed.

A PDE is classified as **degenerate elliptic** if

$$F(x, r, p, X) \leq F(x, r, p, Y), \quad \text{whenever} \quad Y \leq X. \tag{4.4}$$

For the Monge-Ampère specifically, the PDE is said to degenerate at any point x where $\det(D^2u(x)) = 0$. Crandall et al express the Monge-Ampère equation as

$$F(x, r, p, X) = \begin{cases} -\det(X) + f(x, r, p), & \text{if } X \geq 0 \\ +\infty, & \text{otherwise} \end{cases} \tag{4.5}$$

where $u(x)$ is convex [29]. This form of the Monge-Ampère equation is degenerate elliptic. It is known that u is convex if and only if $D^2u(x) \geq 0$. The $+\infty$ condition is enforced for the sake of ellipticity. Degenerate elliptic operators such as the one in Equation (4.5) may have solutions that are not differentiable. In this case, the derivative terms Du and D^2u would be undefined. We must define a more general class of functions which satisfy the $F = 0$ in a weaker sense.

Before defining the weak solution, let us define semicontinuity. A function $f : \Omega \to \mathbb{R}$ is **upper** (or respectively **lower**) **semicontinuous** at a point $x_0 \in \Omega$ if

$$f(x_0) \geq \limsup_{x \to x_0} f(x) \tag{4.6}$$

(or respectively $f(x_0) \geq \liminf_{x \to x_0} f(x)$). Let X be a topological space. The **upper** (or respectively **lower**) **semicontinuous envelope** of a function $F : X \to \mathbb{R}$ which we denote by F^* (or F_*) is defined to be the smallest (respectively largest) upper (respectively lower) semicontinuous function G such that $G \geq F$ (respectively $G \leq F$).

Next, we define the viscosity solution. A bounded upper (or lower) semicontinuous function u is a **viscosity subsolution** (or respectively **supersolution**) of $F = 0$ if for every $\phi \in C^2(\overline{\Omega})$ and all x in $\overline{\Omega}$ such that $u - \phi$ has a local maximum

(respectively minimum) at x, then we have

$$F_*^{(*)}\left(x, u(x), \nabla\phi(x), D^2\phi(x)\right) \leq (\geq) 0. \tag{4.7}$$

A bounded function $u : \overline{\Omega} \to \mathbb{R}$ is called a **viscosity solution** to the problem $F = 0$ if it is both a viscosity subsolution and supersolution to $F = 0$.

The definition of the viscosity solution does not require u to be smooth because the derivative operations are imposed on the test functions. To see that viscosity solutions are an appropriate generalization, consider if u was smooth. Let $\hat{x}$ be a local maximum of $u - \phi$. Then we have

$$F(\hat{x}, u(\hat{x}), D\phi(\hat{x}), D^2\phi(\hat{x})) \leq F(\hat{x}, u(\hat{x}), Du(\hat{x}), D^2u(\hat{x})) \leq 0$$

because $Du = D\phi$ and $D^2u \leq D^2\phi$. A viscosity subsolution is also a classical subsolution.

The comparison principle is central to nonlinear elliptic equations. Let u be an upper semicontinuous subsolution of $F \leq 0$ in Ω, and v a lower semicontinuous supersolution of $F \geq 0$. The PDE satisfies a **comparison principle** if $u \leq v$ on $\partial\Omega$ implies $u \leq v$ throughout $\bar{\Omega}$. This principle provides a route to uniqueness of viscosity solutions [29].

4.1.1 Dirichlet Problem

The BVP of our focus is the Dirichlet Monge-Ampère problem expressed as follows

$$\begin{cases} \det{}^+(D^2u(x)) = f(x, u(x), Du(x)) & \text{if } x \in \Omega \\ u(x) = g(x) & \text{if } x \in \partial\Omega \end{cases} \tag{4.8}$$

for $f > 0$ on Ω. The domain Ω must be a convex subset of $\mathbb{R}^n$. The modified operator $\det^+$ can be any appropriate operator that satisfies the anti-monotonicity property.

In our context, it will be defined as

$$\det{}^{+}(M) = \begin{cases} \det(M), & M \geq 0 \\ \lambda_1, & \text{otherwise,} \end{cases} \tag{4.9}$$

where $M \geq 0$ denotes that M is positive semidefinite and λ_1 is the smallest eigenvalue of M. The Dirichlet data $g(x)$ is continuous.

The existence and uniqueness of the viscosity solution depend on the form of f. In this dissertation, we consider two cases for the right-hand side. The first is

$$f(x, u(x), Du(x) = \kappa(x) \left(1 + |Du(x)|^2\right)^{(n+2)/2}. \tag{4.10}$$

The nonnegative function $\kappa : \Omega \to [0, \infty)$ prescribes the Gaussian curvature of the surface u. In the second case, f is independent of u and Du, so $f = f(x)$. Well-posedness results covering the cases considered here are established in [6, 24, 39].

4.2 Discretization

The main technique for approximating the Monge-Ampère operator is the finite difference method. The golden rule of finite schemes for linear PDEs is:

$$\text{consistent} + \text{stable} \iff \text{convergence.}$$

Consistency and stability alone are insufficient for the nonlinear Monge-Ampère equation. For example, approximating $u_{xy}u_{yx}$ by a product of second-order centered differences gives a consistent scheme without ensuring convergence. A nonlinear convergence framework for viscosity solutions requires an additional condition.

We begin with the Barles–Souganidis framework for convergent finite difference schemes. Under its comparison assumptions, the central implication is

$$\text{consistent} + \text{stable} + \text{monotone} \Rightarrow \text{convergence.}$$

Next, we discuss wide stencil schemes tailored to the Monge-Ampère operator using second directional derivatives. Finally, we present a scheme based on an integral representation of $\det(D^2u)$.

4.2.1 Monotone Discretization

Barles and Souganidis established sufficient conditions for convergence of approximation schemes for nonlinear second-order PDEs in 1991 [7]. Under the relevant comparison assumptions, a monotone, stable, consistent scheme converges to the viscosity solution of Equation (4.3). We now define these conditions.

Let Ω be a bounded computational domain and Ω^h a finite set of grid points discretizing it. Write the discrete form of Equation (4.3) at $x \in \Omega$ as

$$F^h(x, u(x), u(x) - u(\cdot)) = 0 \tag{4.11}$$

where $u(\cdot)$ is an evaluation of u at some point in Ω^h. The argument, $u(x) - u(\cdot)$, is understood to be a shorthand for the collection of all dependency on $u(x)$ minus u evaluated at some grid point in Ω^h. For example, if $F = -\frac{d^2}{dx^2}u(x)$ and F^h is discretized with the standard 1D centered difference on a uniform grid, then the third argument collectively includes both $u(x) - u(x+h)$ and $u(x) - u(x-h)$.

A scheme of the form shown in Equation (4.11) is said to be **monotone** if F^h is a non-decreasing function in its last two arguments. The monotonicity can be spotted most easily in linear stencils. The simplest example is the first-order backward difference method

$$u' \approx \frac{u(x) - u(x-h)}{h}.$$

The stencil is monotonic because $h > 0$. The classic centered difference formula for $F = -u''(x)$,

$$-u'' \approx \frac{-u(x+h) + 2u(x) - u(x-h)}{h^2},$$

is also monotonic. The author of [57] gives more examples of how monotonic schemes can be constructed by using order-preserving operators such as the max and min functions.

A scheme of the form in Equation (4.11) is said to be **stable** if solutions to Equation (4.11) are uniformly bounded with respect to h. Precisely, there exists some $M > 0$ independent of h such that any bounded solution u^h of Equation (4.11) satisfies $\left\|u^h\right\|_\infty < M$.

A scheme of the form in Equation (4.11) is said to be **consistent** if for any test function $\phi \in C^{2,1}(\bar{\Omega})$ and $x \in \bar{\Omega}$, we have

$$\begin{aligned} \limsup_{h\to 0^+, y\to x, \xi\to 0} F^h(y, \phi(y)+\xi, \phi(y)-\phi(\cdot)) &\leq F^*\left(x, \phi(x), \nabla\phi(x), D^2\phi(x)\right) \\ \liminf_{h\to 0^+, y\to x, \xi\to 0} F^h(y, \phi(y)+\xi, \phi(y)-\phi(\cdot)) &\geq F_*\left(x, \phi(x), \nabla\phi(x), D^2\phi(x)\right). \end{aligned} \tag{4.12}$$

This definition is equivalent to saying that the local truncation error goes to zero as $h \to 0$.

The original Barles–Souganidis result assumes a strong comparison principle for $F = 0$ [7]. The Dirichlet Monge-Ampère formulation in Equation (4.8) does not directly satisfy that assumption. Hamfeldt and Oberman developed convergence arguments using discrete comparison principles. Convergence results and guidelines for constructing monotone Monge-Ampère discretizations are given in [39, 45].

4.2.2 Wide Stencil Methods

The convergence criterion is powerful, but constructing a monotone scheme for a nonlinear PDE can be difficult. Standard finite difference approximations of second derivatives have accuracy $O(h^2)$ [57]. For some linear degenerate elliptic PDEs, even a monotone $O(h^2)$ scheme cannot be constructed [52].

The Monge-Ampère equation has a distinct advantage in that the determinant operator can be written in terms of eigenvalues. There are several benefits to this.

One, a discretization is naturally introduced because in $\mathbb{R}^n$ the determinant is a product of n eigenvalues. Two, we may use the classical Rayleigh-Ritz method to write the eigenvalues in terms of angular derivatives [57]. Consider the 2D Monge-Ampère equation. We may write the determinant as

$$\begin{aligned} \det D^2 u(x) &= \lambda_1(x)\lambda_2(x) \\ &= \min_{\theta}(u_{\theta\theta}(x)) \max_{\theta}(u_{\theta\theta}(x)), \end{aligned} \tag{4.13}$$

where $u_{\theta\theta} = \frac{d^2u}{d\nu_\theta^2}(x)$ for $\nu_\theta = (\cos\theta, \sin\theta)$. This formulation is powerful because we are free to choose discrete set of angles $0 \leq \theta_0 < \theta_1 < \ldots < \theta_M < \pi$ and approximate Equation (4.13) as

$$\det D^2 u(x) \approx \min_{i}(u_{\theta_i\theta_i}(x)) \max_{i}(u_{\theta_i\theta_i}(x)). \tag{4.14}$$

The θ_i are reference angles which can be chosen to line up with neighboring grid points. Furthermore, the second directional derivative $u_{\theta\theta}$ have monotone finite difference approximations. This style of discretization is referred to as the **wide stencil method** [52].

The second directional derivatives can be discretized with the monotone scheme

$$\frac{d^2u}{d\nu^2}(x) \approx 2\frac{h^- u\left(x + h^+\nu\right) + h^+ u\left(x - h^-\nu\right) - \left(h^+ + h^-\right) u(x)}{h^+h^-\left(h^+ + h^-\right)}, \tag{4.15}$$

where h^+ and h^- are the forward and backward step sizes in the direction ν. The name "wide stencil" comes from the fact that the sum $h^+ + h^-$ is typically larger than $2h$, where h is the spatial resolution of the mesh. Thus, the wide stencil method has a stencil pattern which looks like spokes of a wheel. See Figure 4.1. The reason why the stencil points are far apart is because the angular resolution $d\theta = max_i|\theta_i - \theta_{i+1}|$ must be made sufficiently small. Figure 4.1 shows how refining the angular resolution also widens the stencil.

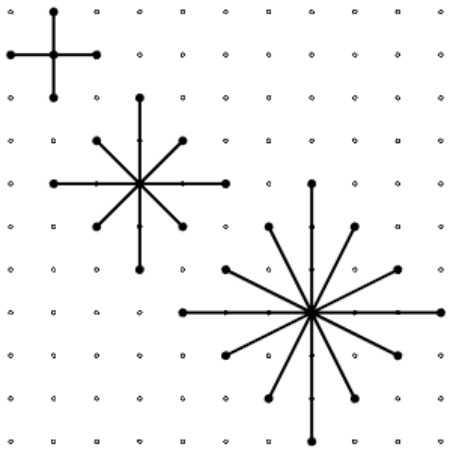


Figure 4.1 Example of a wide stencil with width levels of 1, 2, and 3.

Wide stencil methods have several further advantages. The max and min operators preserve order, so combining Equation (4.15) with Equation (4.14) yields a monotone scheme. Stencil points outside the computational domain can be projected onto its boundary while preserving monotonicity. Increasing the stencil width increases the work per evaluation, but at most linearly in the stencil size. See [41, 52].

4.2.3 Determinant Integral Representation Quadrature

This section summarizes the wide stencil method of [22], based on an integral representation of the Hessian determinant. Quadrature over the angular variable produces a discrete Monge-Ampère operator.

It can be shown that in two dimensions $\Omega \subset \mathbb{R}^2$ $u \in C^2(\Omega)$, if u is strictly convex then

$$\det(D^2 u(x)) = \left(\frac{1}{\pi} \int_0^\pi \frac{d\theta}{u_{\theta\theta}(x)} \right)^{-2}. \tag{4.16}$$

This integral can be discretized with a quadrature over the angle $0 < \theta < \pi$. Doing so will yield a discrete form of the Monge-Ampère operator. In practice, however, the convexity of the solution is enforced by making a slight modification to the determinant operator. Consider the relaxed, convexified version of the Monge-Ampère

operator

$$\det_{\varepsilon}^{+}\left(D^2u(x)\right) = \left(\frac{1}{\pi}\int_0^{\pi}\frac{d\theta}{\max\left(u_{\theta\theta}(x),\varepsilon\right)}\right)^{-2} + \min_{\theta\in[0,\pi)}\left\{\min\left(u_{\theta\theta},\varepsilon\right)\right\}. \tag{4.17}$$

The regularization parameter $\epsilon > 0$ avoids singularities when $\det(D^2u(x)) = 0$ and in the associated Jacobian; see [41]. Regularization introduces error, but ϵ can be chosen small enough that spatial discretization error dominates, preserving consistency [22].

Applying a quadrature to 4.17 yields

$$\det_{\varepsilon}^{+}(D^2u(x)) \approx \left(\frac{1}{\pi}\sum_{i=0}^{M}\frac{w_i}{\max\left\{\mathcal{D}_{\theta_i\theta_i}u(x),\epsilon\right\}}\right)^{-2} + \min_{i=0,\ldots,M}\left\{\mathcal{D}_{\theta_i\theta_i}u(x),\epsilon\right\} \tag{4.18}$$

where $w_i > 0$ are quadrature weights satisfying $\sum_i^M w_i = \pi$, $0 \le \theta_0 < \ldots < \theta_M < \pi$ are the discrete angles, and $\mathcal{D}_{\theta_i\theta_i}u(x)$ approximates the second derivative in direction $\nu_{\theta_i} = (\cos\theta_i, \sin\theta_i)$. Monotonicity of the directional derivative approximation ensures monotonicity of the combined scheme. Under mild quadrature assumptions, the local truncation error is bounded by contributions from finite differences, quadrature, and regularization [22], including when Hessian eigenvalues vanish.

The second directional derivative will be discretized with the monotonic finite difference scheme:

$$\begin{aligned}\mathcal{D}_{\theta_i\theta_i}u(x) &\equiv 2\frac{h^-u\left(x+h^+\nu_{\theta_i}\right)+h^+u\left(x-h^-\nu_{\theta_i}\right)-\left(h^++h^-\right)u(x)}{h^+h^-\left(h^++h^-\right)}\\ &= u_{\theta_i\theta_i}(x)+\mathcal{O}\left(h^+-h^-\right)+\mathcal{O}\left(\frac{\left(h^+\right)^3+\left(h^-\right)^3}{h^++h^-}\right).\end{aligned} \tag{4.19}$$

where the step sizes $h^+, h^- > 0$ are chosen so that stencil points $x \pm h^{\pm}\nu_{\theta_i}$ lie at the grid points in Ω^h. The h^+ is the forward step size in the direction ν_{θ_i}, and h^- is the backward step size in the direction $-\nu_{\theta_i}$. The step sizes actually depend on x. Additionally, h^+ and h^- need to be equal. For certain $x \in \Omega$, if either stencil point $x \pm h^{\pm}\nu_{\theta_i}$ lie outside of the computational domain, they must be projected onto the boundary $\partial\Omega$ in a way that preserves the reference angle θ_i. The quantity

$r = \max_x(h^{\pm}(x))$, the largest stencil point distance, is called the **stencil width** of the scheme. The scheme described thus far is general enough to work on any mesh. Next, we give an example of how to implement this wide stencil scheme on a Cartesian mesh.

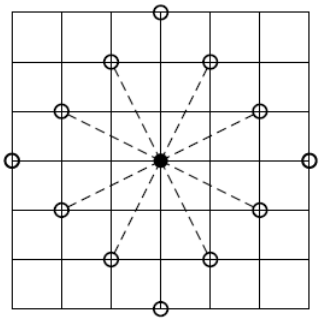


Figure 4.2 The L^1 stencil with $d = 3$. This figure was taken from [22].

Suppose Ω^h is a uniform Cartesian mesh in 2D so that the nodal solution can be represented as $u_{i,j}$ where i, j are positive indices corresponding the row and column. To implement a wide stencil method, an algorithm is needed to choose the angles and systematically compute the step sizes at any point based on a desired stencil width r. The method proposed in [22] is the L1 stencil. We assume a stencil width of the form $r = dh$, where d is a positive integer. Let $\mathbb{Z}^2$ be the set of lattice points, let us define the L^1 norm for $z = (z_1, z_2) \in \mathbb{Z}^2$ to be

$$\|z\|_{L^1} = |z_1| + |z_2|. \tag{4.20}$$

For the L1 stencil of width $r = dh$, we choose the stencil points for $u_{i,j}$ to be the set of all index pairs $(i + z_1, j + z_2)$ satisfying

$$\{\|(i, j) + z\|_{L^1} = d \ \mid z = (z_1, z_2) \in \mathbb{Z}^2\}. \tag{4.21}$$

The set 4.21 contains $4d$ points. Figure 4.2 shows an example of the L1 stencil with $d = 3$.

A stencil can be generated for any d, and the choice naturally induces the angles θ_i. For a given d, we have $M = 2d - 1$ with $\theta_0 = 0$. The remaining angles θ_i needed for the quadrature can be determined with basic trigonometry. It was proven that the L1 stencil is consistent [22].

The overall error of the method depends on the spatial resolution, the angular resolution, and the regularizing parameter. The main discretization parameter is h. The truncation error for the finite difference is like $\tau_{FD} = O(r^2)$ when away from the boundary. The truncation error for the Simpson's rule is like $\tau_\theta = O(d\theta^4)$. It turns out that $d\theta = O(\frac{h}{r})$. The truncation error for the overall scheme is

$$\begin{aligned} \tau &= O(\tau_{FD} + \tau_\theta + \epsilon) \\ &= O(r^2 + d\theta^4 + \epsilon) \\ &= O\left(d^2h^2 + \frac{h^4}{(dh)^4} + \epsilon\right). \end{aligned} \tag{4.22}$$

All three terms must vanish as h tends to zero. To limit computational cost, choose the smallest stencil width that balances the finite difference and quadrature errors. Setting $\tau_{FD} = \tau_\theta$ gives $d = O(h^{-1/3})$. With Simpson's rule, choosing $d = \lceil h^{-1/3} \rceil$ yields overall accuracy $\tau = O(h^{4/3})$. A larger regularization parameter can accelerate the solve but increases error; the choice $\epsilon = r^2$ balances these effects without reducing the stated accuracy.

4.3 Nonlinear Methods and Domain Decomposition

This section describes nonlinear solvers and their combination with domain decomposition. We begin with Newton's method, then discuss compositions of solvers and parallel variants.

The nonlinear additive Schwarz method (NASM) is a domain decomposition algorithm which also falls into this category [20]. NASM can be thought of as an ad-hoc adaptation of the classic Schwarz method for nonlinear BVPs. NASM does

not have to be used as a global method; it can instead be used a preconditioner for other nonlinear methods. An alternative way to introduce domain decomposition to an nonlinear problem is to use it as a linear preconditioner in Newton's method. Newton's method requires a linear system to be solved at each iteration. This linear system can be parallelized by using a Schwarz preconditioner.

4.3.1 Newton's Method

We are interested in numerical methods for solving discrete problems of the form

$$\mathbf{F}(\mathbf{u}) = \mathbf{b} \tag{4.23}$$

where $\mathbf{F} : \mathbb{R}^n \to \mathbb{R}^n$ can be any nonlinear expression. In the context of BVPs, $\mathbf{F}$ is the discrete form of a nonlinear PDE, $\mathbf{u}$ is the n-dimensional approximation of the solution $u(x)$ at the discrete grid points Ω^h, and $\mathbf{b}$ is an appropriate discrete form of the source function and boundary data. The goal of nonlinear algorithms is to iteratively reduce the backward error usually defined to be the norm of the **residual**

$$\mathbf{r}(\mathbf{u}) = \mathbf{F}(\mathbf{u}) - \mathbf{b}. \tag{4.24}$$

The most prominent method to solve Equation (4.23) is the Newton-Raphson (Newton's) method. Newton's method plays a role in almost every nonlinear solver. Starting with a suitable initial guess $\mathbf{u}_0$, one step of **Newton's method** proceeds as follows:

Algorithm 6: Newton's Method

Data: $\mathbf{u}_i$
Result: $\mathbf{u}_{i+1}$
Solve $\mathbf{J}(\mathbf{u}_i)\mathbf{d} = \mathbf{r}(\mathbf{u}_i)$;
Compute $\operatorname{argmin}_\lambda \|\mathbf{r}(\mathbf{u}_i + \lambda\mathbf{d})\|$; `/* line search */`
$\mathbf{u}_{i+1} = \mathbf{u}_i + \lambda_i\mathbf{d}$;

The $\mathbf{d}$ is the step direction and the $\lambda > 0$ the step magnitude. The vector $\mathbf{d}$ is a solution to a linear system involving the Jacobian of $\mathbf{F}$ which must be solved each iteration. The **Jacobian matrix** is defined in the usual way:

$$(\mathbf{J}(\mathbf{u}))_{ij} = \frac{\partial F_i}{\partial u_j}, \tag{4.25}$$

where F_i and u_j are components of $\mathbf{F}$ and $\mathbf{u}$. Because the Jacobian changes with $\mathbf{u}_i$, repeatedly factoring it can be expensive. Approximating $\mathbf{d}$ with a preconditioned Krylov method gives a **Newton-Krylov** method. This term covers a broad family of solvers with different linear preconditioners and stopping criteria.

Newton's method in optimization is often paired with line search methods to improve the stability. A line search algorithm, such as backtracking (BT) or the L2, is used to compute the scaling factor λ to adjust the Newton step size. The exact algorithms for the BT and L2 line search can be found in [20]. Given a direction $\mathbf{d}$, the goal of line search is to iteratively decrease $\|\mathbf{r}(\mathbf{u}_i + \lambda\mathbf{d})\|$ with respect to λ. For instance, the L2 line search is based on finding the minimum of $\|\mathbf{r}(\mathbf{u}_i + \lambda\mathbf{d})\|_2^2$ with a secant search. The (BT) line search iterates until a pair of inequalities called Wolfe conditions are met. The advantage of BT is that it often defaults to taking the full step because one of the Wolfe conditions is already satisfied. When the BT does not default, however, the algorithm is more costly than the L2 line search.

4.3.2 Nonlinear Preconditioning

Newton's method falls into a broader class of nonlinear solvers. We define a **nonlinear stationary solver $\mathbf{M}$** to be an operator that can be used to iterate the solution of Equation (4.23) in the form

$$\mathbf{u}_{i+1} = \mathbf{M}(\mathbf{u}_i\,; \mathbf{F}, \mathbf{b}) \tag{4.26}$$

In a sense, $\mathbf{M}$ is some procedure which produces a new solution which lowers $\|\mathbf{r}(\mathbf{u})\|$ using an initial guess $\mathbf{u}_i$. Suppose $\mathbf{M}$ and $\mathbf{N}$ are both stand-alone stationary solvers. The simplest way to combine the two is either additively or multiplicatively. They can be written formally as

$$\mathbf{u}_{i+1} = \mathbf{M}\left(\mathbf{u}_i\,;\mathbf{F},\mathbf{b}\right) + \mathbf{N}\left(\mathbf{u}_i\,;\mathbf{F},\mathbf{b}\right), \quad \text{and} \tag{4.27}$$

$$\mathbf{u}_{i+1} = \mathbf{M}\left(\mathbf{N}\left(\mathbf{u}_i\,;\mathbf{F},\mathbf{b}\right)\,;\mathbf{F},\mathbf{b}\right). \tag{4.28}$$

Additive composition can use weighted coefficients, while multiplicative composition nests one solver inside another. These are examples of **composing** nonlinear solvers [20]. Preconditioning provides further possibilities.

Most of the nonlinear solvers have three basic elements: defining the quantity to minimize, computing the search direction, and calculating the update. The quantity to minimize is usually the norm of the residual defined in Equation (4.24). Replacing the residual for $\mathbf{M}$ with

$$\mathbf{r}(\mathbf{u}) = \mathbf{u} - \mathbf{N}(\mathbf{u}\,;\mathbf{F},\mathbf{b}) \tag{4.29}$$

is referred to as **nonlinear left-preconditioning**. The preconditioned solver can be written compactly as

$$\mathbf{u}_{i+1} = \mathbf{M}\left(\mathbf{u}_i\,;\mathbf{u}_i - \mathbf{N}\left(\mathbf{u}_i\,;\mathbf{F},\mathbf{b}\right),\mathbf{b}\right). \tag{4.30}$$

Nonlinear right preconditioning is similar to the linear case. The original problem is $\mathbf{F}(\mathbf{u}) = \mathbf{b}$. We may introduce an auxiliary unknown $\mathbf{u} = \mathbf{N}(\mathbf{y};\mathbf{F},\mathbf{b})$ which in a sense means that $\mathbf{y}$ is one iteration away from $\mathbf{u}$. The **nonlinear right-preconditioning** with $\mathbf{N}$ of the algorithm $\mathbf{u}_{i+1} = \mathbf{M}(\mathbf{u}_i\,;\mathbf{F},\mathbf{b})$ is defined to be the two-step algorithm:

$$\begin{cases} \mathbf{y}_{i+1} = \mathbf{M}(\mathbf{y}_i\,;\mathbf{F}(\mathbf{N}(\cdot)),\mathbf{b}) \\ \mathbf{u}_{i+1} = \mathbf{N}(\mathbf{y}_{i+1}\,;\mathbf{F},\mathbf{b}). \end{cases} \tag{4.31}$$

We have described four ways to compose two nonlinear solvers. The next section introduces a stationary solver based on domain decomposition.

4.3.3 Nonlinear Additive Schwarz

The nonlinear additive Schwarz method (NASM) is another example of nonlinear solver introduced in [20]. The method is stated in Algorithm 7.

Algorithm 7: NASM

Data: $\mathbf{u}_i$
Result: $\mathbf{u}_{i+1}$
for $B = 1 : N_B$ **do**
  $\mathbf{u}_0^B = \mathbf{R}^B \mathbf{u}_i$;
  $\mathbf{r}_0^B = \mathbf{R}^B \mathbf{r}_i(\mathbf{u}_i)$;
  $\mathbf{u}^B = \mathbf{M}^B(\mathbf{u}_0^B; \mathbf{F}^B, \mathbf{r}_0^B)$;
end
$\mathbf{u}_{i+1} = \mathbf{u}_i + \sum_{B=1}^{N_B} \mathbf{P}^B(\mathbf{u}_0^B - \mathbf{u}^B)$

NASM solves subdomain problems simultaneously and combines their solutions. Boundary data are supplied by the previous global iterate $\mathbf{u}_i$. Restriction and projection operators formalize this construction. Consider N_B sub-vectors of $\mathbf{u}$, $\{\mathbf{u}^1, \ldots, \mathbf{u}^{N_B}\}$, whose elements need not be unique. The **restriction operator** for the Bth sub-vector, denoted by $\mathbf{R}^B$, is defined to be a Boolean matrix satisfying $\mathbf{u}^B = \mathbf{R}^B \mathbf{u}$. Let $\mathbf{F}^B$ be an appropriate restriction of $\mathbf{F}^B(\mathbf{u}^B)$ on the sub-vector. The $\mathbf{M}^B$ is any nonlinear solver and $\mathbf{P}^B$ is a projection operator which transmits the local solution back to the global vector. It is common to choose $P^B = (R^B)^T$ which means the NASM is a restrictive type.

NASM introduces parallelism by combining local solutions, but this alone provides little global guidance. Consequently, convergence is not guaranteed for a general nonlinear problem [20].

NASM is already a composition: it uses a nonlinear solver $\mathbf{M}^B$ on each block B. NASM is the outer solver, and $\mathbf{M}^B$ is the inner solver. Either can be preconditioned, and NASM can itself serve as a preconditioner. Different blocks may use different local solvers. Choosing Newton-Krylov locally gives the Schwarz-Newton-Krylov method considered in Chapter 5.

The local solve accuracy must be chosen carefully. More accurate local solutions may reduce the number of NASM iterations and communication steps, but can waste work before neighboring data are accurate. Less accurate local solves exchange information more frequently at lower cost per iteration. Excessive under-solving can cause stagnation or divergence. The appropriate balance depends on the problem and the desired robustness.

4.3.4 Nonlinear Preconditioning with Domain Decomposition

NASM is a nonlinear solver based on the principles of domain decomposition. Therefore any composition of solvers which involves NASM should be considered a nonlinear DDM. Some of the nonlinear solvers are: Newton's method, quasi-Newton, nonlinear GMRES (NGMRES), nonlinear conjugate gradient (NCG), and full approximation scheme (FAS). Each of these methods are described in [20].

Let us begin with Newton's method preconditioned with NASM. The earliest example which features this type of composition is the additive Schwarz preconditioned inexact Newton (ASPIN) algorithm proposed in [26]. The systems arising from computational fluid dynamics can be very poorly conditioned. Therefore, the NASM is as a preconditioner to improve the robustness of the inexact Newton' method. **Inexact Newton** is a variant of Newton's method in which the solution to the Jacobian system $\mathbf{J}(\mathbf{u}_i)\mathbf{d} = \mathbf{r}(\mathbf{u}_i)$ is relaxed. Given some tolerance $\eta > 0$, the goal is to find $\mathbf{d}$ satisfying $\|\mathbf{r}(\mathbf{u}_i) - \mathbf{J}(\mathbf{u}_i)\mathbf{d}\| \leq \eta \|\mathbf{r}(\mathbf{u}_i)\|$. In a more recent work [33], the authors propose the restricted additive Schwarz preconditioned exact Newton (RASPEN) as

an alternative to ASPIN. It is a preconditioning of the exact Newton algorithm and it is demonstrated to outperform the ASPIN. If large Newton iterations becomes an issue in the Monge-Ampère equation, then the RASPEN could be a possible fix.

The idea behind NGMRES is to memorize all of the previous solutions $\mathbf{u}_j$ for $i = 1, 2, \ldots, i$ and use them to compute the next iteration $\mathbf{u}_{i+1}$. One of its drawbacks is that it uses a crude method for computing the search direction. It simply takes $\mathbf{d}$ to be the negative gradient of $\|\mathbf{r}(\mathbf{u})\|$. This is the same as the nonlinear Richardson algorithm (a different nonlinear solver). Both the NGMRES and nonlinear Richardson fail to converge for the Monge-Ampère equation. However, if the NGMRES was preconditioned from the left with NASM, then the NGMRES will converge. The composition is a unique algorithm which remembers all of the subdomain solutions.

The FAS is an example of a nonlinear multigrid method. It is classified as a decomposition algorithm just like the NASM. It requires a nonlinear solver to compute a solution on smaller portion of the original problem. It lacks a globalization strategy. Therefore, it may be best to combine the FAS additively with the NASM. It is common practice for linear DDMs to incorporate one coarse mesh as part of the many subdomains decompositions [63]. The combination NASM + FAS is a nonlinear analog to the coarse-corrected DDM. The combination with FAS may result in fewer NASM iterations than if the NASM was used alone.

4.3.5 Newton-Krylov-Schwarz

Domain decomposition can also enter through linear preconditioning, without an outer NASM iteration. Newton-Krylov solves a Jacobian system at every nonlinear iteration. Preconditioning that system with the additive Schwarz method gives **Newton-Krylov-Schwarz** [25], a parallel form of Newton-Krylov.

Let us discuss the Krylov-Schwarz aspect of the algorithm. In the NKS algorithm, the Jacobian system is preconditioned from the right with the ASM. Consider

the linear system $\mathbf{Ax} = \mathbf{b}$ and the preconditioner $\mathbf{M}^{-1}$. The right-preconditioned GMRES is given in Algorithm 8.

Algorithm 8: Right-preconditioned GMRES

Data: Guess $\mathbf{x}_0$, restarting parameter m
Result: $\mathbf{x}_{final}$
$\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$;
$R = \|\mathbf{r}_0\|_2$;
$\mathbf{v}_1 = \mathbf{r}_0/R$;
for $j = 1, 2, \ldots, m$ **do**
  $\mathbf{z}_j = M^{-1}\mathbf{v}_j$; `/* apply preconditioner */`
  $\mathbf{w} = A\mathbf{z}_j$;
  for $i = 1, \ldots, j$ **do**
    $h_{i,j} = (\mathbf{w}, \mathbf{v}_i)$;
    $\mathbf{w} = \mathbf{w} - h_{i,j}\mathbf{v}_i$;
  end
  $Z = [\mathbf{z}_1, \ldots, \mathbf{z}_m]$;
end
$\mathbf{y}_m = \text{argmin}_y \|R\mathbf{e}_1 - H\mathbf{y}\|_2$; `/* Use QR for least squares */`
$\mathbf{x}_m = \mathbf{x}_0 + Z\mathbf{y}_m$;
Repeat if necessary with $\mathbf{x}_0 = \mathbf{x}_m$;

The basic **additive Schwarz method** preconditioner for the linear problem $Ax = b$ has the form

$$M^{-1} = \sum_{i=1}^{N_d} P_i A_i^{-1} R_i, \tag{4.32}$$

where R_i for $i = 1, \ldots, N_d$ is an appropriate set of restriction operators. The projection operator P_i holds information about how the local solutions contribute to the global solution. The A_i corresponds to the local block $A_i = R_i A R_i^T$. The matrix M^{-1} is never actually constructed in practice. Applying a Schwarz preconditioner to compute $M^{-1}\mathbf{v}$ is given in Algorithm 9.

Algorithm 9: Applying the additive Schwarz preconditioner

Data: $\mathbf{v}$
Result: $\mathbf{u} = M^{-1}\mathbf{v}$
Scatter $\mathbf{v}$ to $\mathbf{v}_j, \forall j$;
for $j = 1, 2, \ldots, N_d$ **do**
  Approximate $\mathbf{u}_j = A_i^{-1}\mathbf{v}_j$; `/* Local parallel solve */`
end
$\mathbf{u} = \sum_i^{N_d} P_i \mathbf{u}_i$; `/* Parallel reduction or gather */`

Applying the additive Schwarz preconditioner requires local solves. These are smaller than the original system $A^{-1}\mathbf{v}$ and can be performed in parallel. To control cost, they may be approximated by incomplete LU (ILU), which limits fill-in by dropping selected entries [27], or by a small number of Krylov iterations.

There are a few variations of the ASM which depend on the choice of P_i. The additive Schwarz usually has subdomains with some overlap. In each iteration, the local information on the overlap can be discarded or recycled in some form of interpolation. The variation we focus on is the restrictive additive Schwarz method (RASM) in which the information is discarded. This means that we let $P_i = R_i^T$. We may still use the additive Schwarz when the intersection between subdomains is just an interface. In this case, the additive Schwarz is equivalent to the block Jacobi preconditioner. This method is not to be confused with the non-overlapping optimized Schwarz. See [61] for a discussion on the variations of the Schwarz algorithms.

Lastly, we discuss the Krylov subspace solver. There are many Krylov subspace solvers but we mainly consider variations of the generalized minimal residual method (GMRES) invented by Saad [59]. GMRES is notorious for its accumulating memory cost. The original algorithm is required to memorize all search directions because they are basis for the Krylov subspace. The **restarting GMRES** is the first acceptable variant in which after every m steps, the Krylov subspace is reset starting with only the most recent direction. However, restarting can cause the method to

stagnate because information about the smallest eigenvalues is lost. The **deflated GMRES** is a preconditioning technique which transfers the approximations of the eigenvalues to the next subspace [37]. There is a variant called the **flexible GMRES** (FGMRES) which simply means the right preconditioner M^{-1} is allowed to vary between iterations. Since we are only interested in Schwarz preconditioners, FGMRES will not be discussed any further. There are linear algebra computations within the GMRES algorithm that can be done in parallel. **Communication avoiding GMRES** (CA-GMRES) and the **pipelined GMRES** (PGMRES) are two alternate implementations of the GMRES that utilize parallel computing to reduce the runtime of the algorithm [66].

4.3.6 Convergence of the NASM for the Monge-Ampère Equation

We have introduced two parallel nonlinear solvers. NKS, in Subsection 4.3.5, parallelizes Newton-Krylov through an algebraic DDM preconditioner. NASM, in Subsection 4.3.3, applies Schwarz iteration directly to the nonlinear problem. NKS relies on the convergence of its Newton iteration and linear solves, whereas NASM may fail even when its local problems are solved correctly [20]. For the Monge-Ampère equation, a convergence argument is available for NASM with a suitable discretization. We outline that argument here.

The main idea behind the convergence proof is to argue that the NASM iteration, Algorithm 7, is itself a monotonic mapping when the Monge-Ampère operator is discretized with a proper, monotone scheme. Refer to Subsection 4.2.1 for monotone schemes. Let us define the NASM iteration as

$$\mathbf{u}_{i+1} = \mathbf{N}(\mathbf{u}_i\,;\mathbf{F},\mathbf{b}) \tag{4.33}$$

where $\mathbf{F}$ is a proper, monotonic discretization of the Monge-Ampère operator, $\mathbf{u}_i$ is the ith iteration of the discrete solution on Ω^h, and $\mathbf{b}$ is the subsequent discretization

of the RHS to Equation (4.8). Suppose that the mapping in Equation (4.33) has a unique fixed point $\mathbf{u}^*$. Due to the monotonicity of the map it can be shown that when the NASM is initialized with a super- or sub-solution, the iteration converges to the fixed point (i.e., the solution to the Monge-Ampère equation). Finally, to show that NASM converges to the fixed point with an arbitrary initial guess $\mathbf{u}_0$, we argue that $\mathbf{u}_0$ can always be bounded above and below by a super- and sub-solution, respectively.

The complete proof was being prepared for a separate article at the time of the original dissertation. Here we summarize three main components:

1. The monotonicity of the NASM operator
2. The existence and uniqueness of the fixed point
3. The convergence of sub- and super-solutions

We state the main results and explain the ideas behind their proofs.

The monotonicity of the NASM operator means that it preserves inequalities. More precisely, it means that $\mathbf{u} \leq \mathbf{v}$ on Ω^h implies $\mathbf{N}(\mathbf{u}\,;\mathbf{F},\mathbf{b}) < \mathbf{N}(\mathbf{v}\,;\mathbf{F},\mathbf{b})$ on Ω^h. This statement comes almost directly from the discrete version of the comparison principle. The discrete comparison principle was briefly mentioned in Section 4.2.1. It states that for a proper monotone scheme F,

$$F(x, u(x), u(x) - u(\cdot)) \leq (<) F(x, v(x), v(x) - v(\cdot))$$

implies that $u \leq (<) v$ on Ω^h. Since we assume $\mathbf{F}$ is a monotone scheme, we know the local restriction $\mathbf{F}^B$ is also monotone (B is the index for the NASM block). Hence, the local solver $\mathbf{M}^B$ is monotonic. Since the restrictive NASM takes a combination of the local solution, we naturally expect the NASM solver to also be monotonic.

A fixed point is a discrete solution $\mathbf{u}^*$ satisfying $\mathbf{u}^* = \mathbf{N}(\mathbf{u}^*\,;\mathbf{F},\mathbf{b})$. We know that a fixed point exists because $\mathbf{F}(\mathbf{u}) = \mathbf{b}$ is a well-posed problem. The solution to $\mathbf{F}(\mathbf{u}) = \mathbf{b}$ has to be a fixed point because if the exact solution is used as an initial

guess, then the NASM will feed exact boundary conditions for the local sub-problems which will yield the same solution. Uniqueness means that if $\mathbf{u}^*$ and $\mathbf{v}^*$ are both fixed points then $\mathbf{u}^* = \mathbf{v}^*$. Suppose that any fixed point was also equivalent to the solution of $\mathbf{F}(\mathbf{u}) = \mathbf{b}$. Then by the well-posedness of the problem, all fixed points must be the same. Thus, the fixed point exists and it is unique. There is some subtlety in proving that the fixed point $\mathbf{u}^*$ satisfies $\mathbf{F}(\mathbf{u}^*) = \mathbf{b}$ due to way NASM adds solutions on the overlapping regions. This point is moot in the case of the restrictive NASM.

Recall that a sub-solution (or super-solution) is a discrete solution $\mathbf{u}$ satisfying $\mathbf{F}(\mathbf{u}) \leq (\geq) 0$ on Ω^h. When the NASM iteration is initialized with either a sub- or super-solution, then it will converge to the solution $\mathbf{u}^*$. To see why this is true, let the initial guess $\mathbf{u}^0$ be a super-solution. We need to prove by induction that

$$\mathbf{u}^n \geq \mathbf{u}^{n+1} \geq \mathbf{u}^* \tag{4.34}$$

for all $n \geq 0$. Proving this requires using the discrete comparison principle at the local level and the monotonicity of the NASM operation. The induction statement implies that the NASM iteration is bounded and it converges to $\mathbf{u}^*$. An analogous argument can be made about sub-solutions. This leads to the proof that the NASM will converge with any arbitrary initial guess. Let us define $\mathbf{N}^n(\cdot)$ as the $\mathbf{N}(\cdot, \mathbf{F}, \mathbf{b})$ applied n times. For any $\mathbf{u}^0$, there is always an upper-bound super-solution $\overline{\mathbf{u}}$ and a lower-bound sub-solution $\underline{\mathbf{u}}$. Using the convergence of sub- and super-solutions, we may show that

$$\mathbf{N}^n(\underline{\mathbf{u}}) \leq \mathbf{N}^n(\mathbf{u}^0) \leq \mathbf{N}^n(\overline{\mathbf{u}}), \tag{4.35}$$

for all $n \geq 0$. Taking $n \to \infty$ in Equation (4.35) yields convergence and completes the proof sketch.

CHAPTER 5

DOMAIN DECOMPOSITION METHODS FOR THE MONGE-AMPÈRE EQUATION

This chapter presents numerical results for DDMs applied to the Monge-Ampère equation. The first model is the one-dimensional prescribed Gaussian curvature problem. We compare overlapping and non-overlapping methods, which produce local Dirichlet and Robin problems, respectively.

The second model is the two-dimensional Monge-Ampère equation on a rectangular domain. Source functions are chosen to produce viscosity solutions with different regularity. We compare Newton-Krylov-Schwarz (NKS) and Schwarz-Newton-Krylov (SNK), varying the subdomain overlap, number of subdomains, and domain size.

The first part addresses the 1D problem and the second the 2D problem. Each introduces the model and discretization before presenting numerical results and concluding remarks.

5.1 Domain Decomposition for the One-dimensional Monge-Ampère Equation

We compare overlapping additive Schwarz and non-overlapping optimized Schwarz methods for the 1D Monge-Ampère equation. After describing the model, meshes, and monotone discretizations, we examine both methods through numerical experiments.

5.1.1 One-dimensional Model Problem

Consider the 1D prescribed curvature problem with curvature $\kappa : (a, b) \to [0, \infty)$ with Dirichlet boundary conditions:

$$\begin{cases} u_{xx} = \kappa \left(1 + u_x^2\right)^{3/2}, & x \in (a, b) \\ u = g_a, & x = a \\ u = g_b, & x = b. \end{cases} \tag{5.1}$$

The curvature κ at a point along a curve $f : \mathbb{R} \to \mathbb{R}$ is related to the osculating circle. Given a point P on a curve, there are an infinite number of tangential circles, but there is only one which passes through another point Q on the curve. The **osculating circle** is defined to be the limiting tangential circle at P passing through Q as $Q \to P$. The curvature κ is related to the radius of the osculating circle by $\kappa = \frac{1}{R}$. See Figure 5.1. In the 1D case, the curvature of f is given by $\kappa(x) = f''(x)/(1 + (f')^2)^{3/2}$. For the prescribed curvature problem, the curvature κ is a known parameter, and the goal is determine the curve $u(x)$.

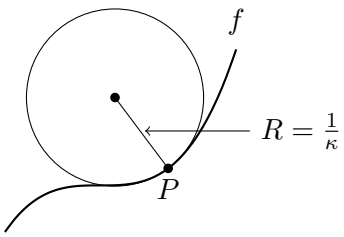


Figure 5.1 The osculating circle of $f(x) = x^3$ at $x = 0.5$.

For the model, we chose the viscosity solution to be the lower right quarter of a unit circle i.e., $a = 0$, $b = 1$, $\kappa = 1$, $g_a = 0$, $g_b = 1$. The viscosity solution is given in closed-form

$$\tilde{u}(x) = -\sqrt{1 - x^2} + 1$$

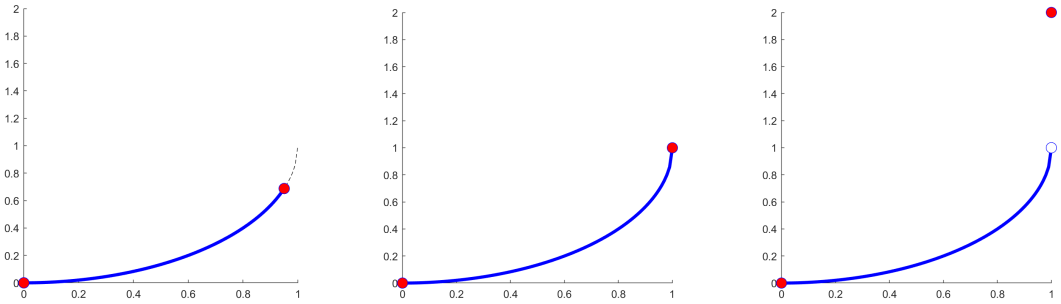
for $x \in (0, 1)$. We note that, under these conditions, the derivative at the right end point $u'(b)$ does not exist. However if we shortened the domain to say $b = 0.99$, then

a classical solution exists. If we chose $g_b = 2$, then our viscosity solution does not change because our circular arc remains to be the largest curve satisfying $u(0) = 0$ with constant curvature. This is the same model discussed in [39].

We are able to manipulate the regularity of the problem by varying the domain or the boundary data. The three cases we consider are

1. When $s = 0.99$ and $g_b = \tilde{u}(s)$
2. When $s = 1$ and $g_b = 1$
3. When $s = 1$ and $g_b = 2$

Figure 5.2 shows the three cases, with prescribed Dirichlet data marked by filled red circles. In case 1, the solution belongs to $C^2(\bar{\Omega})$ and is classical; g_b is chosen to match the exact solution. In case 2, $g_b = \tilde{u}(1) = 1$, and the solution belongs to $C^2(\Omega) \cap C(\bar{\Omega})$ but loses differentiability at the right endpoint. In case 3, $g_b = 2$ and the boundary data are not attained continuously, while $u \in C^2(\Omega)$.



(a) Case 1 **(b)** Case 2 **(c)** Case 3

Figure 5.2 Three cases for the Gaussian curvature problem.

To show the difference in each case, the numerical solution was computed for each case with the Trust-Region-Dogleg algorithm [55]. The grid was uniformly meshed with $h = (b - a)/(N + 1)$. Figure 5.3 shows the pointwise error and the computational runtime for each of the three cases with $N = 2056$ nodes, or step size $h \approx 10^{-4}$. The plot shows the logarithm of the pointwise error $E(x) = |u(x) - u^h(x)|$ over the domain. For all three cases, the error curves show the same behavior until

they near the right boundary point $x = 1$. While case 1 shows a sharp decrease in error at $x = 1$, cases 2 and 3 show a sharp increase in error. The error difference for cases 2 and 3 is about 2 or 3 orders of magnitude worse than case 1. Case 3 also has about a 25% longer runtime than the other two cases.

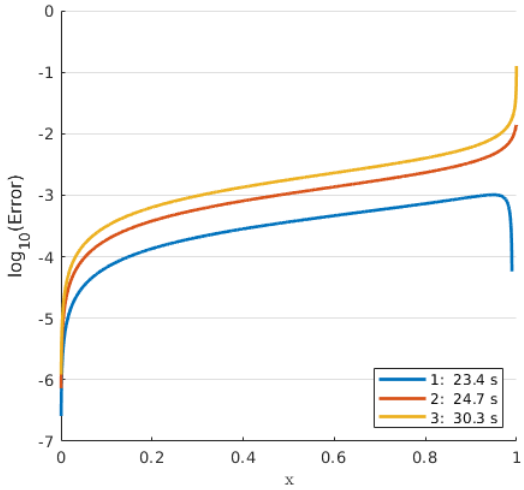


Figure 5.3 Pointwise error and runtime for the three cases.

5.1.2 Non-overlapping Domain Decomposition

In this section, we describe the optimized Schwarz algorithm for the 1D Gaussian curvature problem. For the non-overlapping DDM, the domain $\Omega := (a, b)$ is decomposed into N_d disjoint open intervals such that $\cup_{i=1}^{N_d} \overline{\Omega}_i = \overline{\Omega}$. We call the points separating two neighboring domains the **interface points** $x_{ij} = \partial\Omega_i \cap \overline{\Omega}_j$. We can extend the notation to endpoints $x_{i0} := a$ and $x_{i,N_d+1} = b$. Figure 5.4 shows a diagram of the decomposed grid with $N_d = 3$.

Let u_i and u_j be solutions coming from two adjacent subdomains. On the interface point x_{ij}, we enforce the **continuity conditions**:

$$\begin{cases} u_i = u_j, \\ \partial_x u_i = \partial_x u_j. \end{cases} \tag{5.2}$$

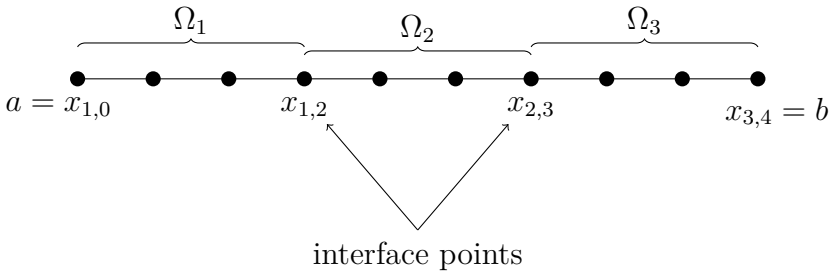


Figure 5.4 Diagram of non-overlapping domain decomposition.

These conditions indicated that the solution to the original problem is at least $C^1(\Omega)$. The derivative term is written as a partial to emphasize that this condition can be generalized to higher dimensions, in which case we would have normal derivatives across a surface. It is standard to rewrite the continuity conditions as the set of Robin boundary conditions:

$$\begin{cases} \beta u_i + \frac{du_i}{dx} = g_{ij}(u_j) := \beta u_j + \frac{du_j}{dx} \\ \beta u_j - \frac{du_j}{dx} = g_{ji}(u_i) := \beta u_i - \frac{du_i}{dx} \end{cases} \tag{5.3}$$

for $1 \leq i, j \leq N_d$, $i \neq j$. The parameter β is the transmission operators which governs the convergence of the DDM. For our problem, β is a scalar parameter whose values must be optimized (hence the name optimized Schwarz).

The **auxiliary unknowns** $g_{ij}(u_j)$ and $g_{ji}(u_i)$ in Equation (5.3) supply boundary data to Ω_i and Ω_j, respectively. Thus g_{ij} is supplied to domain i from domain j. These functions define both the local problems and the interface update. The local PDE operator on Ω_i is

$$F_i\left(x, u_i, \frac{du_i}{dx}, \frac{d^2u_i}{dx^2}; g_{i,i\pm1}\right) = \begin{cases} -\frac{d^2u_i}{dx^2} + \left(1 + \left(\frac{du_i}{dx}\right)^2\right)^{3/2}, & x \in \Omega_i \\ \beta u_i - \frac{du_i}{dx} - g_{i,i-1}, & x = x_{i,i-1} \\ \beta u_i + \frac{du_i}{dx} - g_{i,i+1}, & x = x_{i,i+1} \end{cases} \tag{5.4}$$

for $i = 2, \ldots, N_d - 1$. The formula for the operator is slightly different for the left-most subdomains $i = 1$ and the right-most subdomain $i = N_d$. For F_1, the

boundary condition at $x = a$ is $u_1 - g_a$. For F_{N_d}, the boundary condition on $x = b$ is $u_{N_d} - g_b$. With these changes, we can extend the definition of Equation (5.4) to include $i = 1$ and $i = N_d$.

Starting from provisional interface data, the DDM solves the local PDEs and updates the data using Equation (5.3). With (n) denoting the iteration index, the non-overlapping method is given in Algorithm 10.

Algorithm 10: Non-overlapping DDM

Data: β
Result: u
Initialize $g_{ij}^{(0)} = 0$;
for $n = 1, 2, \ldots$ *until convergence* **do**
  for $i = 1, 2, \ldots, N_d$ **do**
    Solve $F_i\left(x, u_i^{(n)}, \frac{d}{dx}u_i^{(n)}, \frac{d^2}{dx^2}u_i^{(n)}; g_{i,i\pm1}^{(n-1)}\right) = 0$; `/* Local solve */`
    Compute $g_{ij}^{(n+1)} = g_{ij}(u_j^{(n)})$; `/* Compute update */`
  end
end
$u = \sum_{i=1}^{N_d} u_i^{(final)}$;

The convergence of Algorithm 10 depends on the choice of β which does not explicitly appear in any of the steps. It can be shown that the update step in the algorithm can be replaced with the recurrence formula

$$g_{ij}^{(n+1)} = -g_{ji}^{(n)} + 2\beta u_j^{(n+1)}. \tag{5.5}$$

This formula is more efficient than using the definition because it removes the need for derivative approximations. This is a feature that all non-overlapping optimized Schwarz algorithms share. The recurrence formula explicitly shows how the convergence of the DDM depends on the convergence of $g_{ij}^{(n)}$ as $n \to \infty$. This indicates that the residual for the NDDM should depend on g_{ij}. Let $\mathbf{g}^{(n)}$ be the vector of length $2(N_d - 1)$ containing values of $g_{ij}^{(n)}$ for all $1 \leq i, j \leq N_d$, $i \neq j$. Then we

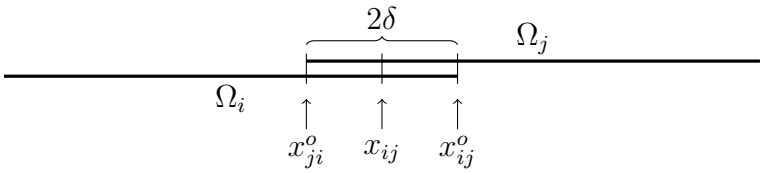


Figure 5.5 Overlapping subdomains.

define the NDDM residual to be

$$r^{(n)} = \left\| \mathbf{g}^{(n)} - \mathbf{g}^{(n-1)} \right\|_2 . \tag{5.6}$$

For the algorithm, choose a tolerance parameter $\epsilon > 0$. We say the algorithm has converged after n iterations when the relative residual satisfies

$$\frac{r^{(n)}}{r^{(1)}} < \epsilon. \tag{5.7}$$

A tolerance of $\epsilon = 10^{-6}$ was used for the 1D DDMs.

5.1.3 Overlapping Domain Decomposition

The overlapping domain decomposition method (ODDM) applies the Schwarz approach to the nonlinear 1D Dirichlet problem. The domain is divided into overlapping subdomains, each with Dirichlet data taken from its neighboring solution.

The subdomains in the ODDM are constructed by introducing interface points $N_d - 1$ artificial interfaces x_{ij} along the interval (a, b) to separate the subdomains. Then an overlap parameter $\delta > 0$ is chosen. The subdomains $\{\Omega_i^o\}, i = 1, ..., N_d$ extend a width δ over the interface points in both directions. See Figure 5.5 for a diagram of the overlap. The interface point of a domain Ω_i^o and its overlapping neighbor Ω_j^o is defined as $x_{ij}^o = \partial\Omega_i^o \cap \overline{\Omega}_j^o$. On these points, we enforce the continuity condition

$$u_i(x_{ij}^o) = u_j(x_{ij}^o) =: g_{ij}, \quad i, j = 1, ..., N_d. \tag{5.8}$$

The notation x_{ij}^o denotes the point where domain i receives data from domain j.

We define the localized PDE operator on Ω_i for the ODDM as

$$F_{i,o}\left(x, u_i, \frac{du_i}{dx}, \frac{d^2u_i}{dx^2}; g_{i,i\pm1}\right) = \begin{cases} -\frac{d^2u_i}{dx^2} + \left(1 + \left(\frac{du_i}{dx}\right)^2\right)^{3/2}, & x \in \Omega_i \\ u_i - g_{i,i-1} & x = x_{i,i-1} \\ u_i - g_{i,i+1}, & x = x_{i,i+1} \end{cases} \tag{5.9}$$

for $i = 1, 2, \ldots, N_d$. The definition of g_{ij} needs to be augmented to include the original data g_a and g_b for $i = 0$ and $i = N_d + 1$. The algorithm for the ODDM is the same as the NDDM stated in Algorithm 10. There are two differences: the auxiliary function g_{ij} and the residual.

There are two ways to interpret the residual. We may use the entire length of the overlap or just the endpoint. If we define

$$\mu^{(n)} = \sum_{i=1}^{N_d-1} \left\| u_i^{(n)} - u_{i+1}^{(n)} \right\|_{\Omega_i \cap \Omega_{i+1}}, \tag{5.10}$$

then an appropriate residual might be $r^{(n)} = |\mu^{(n)} - \mu^{(n-1)}|$. This type of residual will enforce a more strict form of continuity at the cost of increasing the total iterations and residual calculation time.

5.1.4 Discretization

We briefly discuss the monotone discretization for the ODDM and NDDM. We approach the discretization of the 1D derivatives using the order-preserving operators as discussed in [57]. For the second order term, we may use the standard centered difference scheme because it is already monotonic:

$$-\frac{d^2u_i(x)}{dx^2} \approx -\frac{u(x+h) - 2u(x) + u(x-h)}{h^2} \tag{5.11}$$

For the first order derivative, we use

$$\left(\frac{du_i(x)}{dx}\right)^2 \approx \frac{1}{h^2} \max(u(x) - u(x-h), u(x) - u(x+h), 0)^2 \tag{5.12}$$

to make it monotonic [52].

The non-overlapping domain decomposition contains derivatives at the boundary. For the Robin boundary conditions, we apply forward or backward difference depending on the sign. Let $\beta > 0$. Then we discretize the Robin conditions with

$$\begin{cases} \beta u(x_0) - \beta \partial_x u(x_0) - g \approx \beta u(x_0) - \frac{u(x_0+h)-u(x_0)}{h} - g \\ \beta u(x_1) + \beta \partial_x u(x_1) - g \approx \beta u(x_1) + \frac{u(x_1)-u(x_1-h)}{h} - g \end{cases} \tag{5.13}$$

where x_0 denotes the left endpoint of a domain, and x_1 denotes the right endpoint.

The overlapping local problem $F_{i,o} = 0$ is guaranteed to have a unique viscosity solution because it is a Dirichlet problem [39]. The non-overlapping local problem $F_i = 0$, on the other hand, has Robin boundary conditions. On the edge, there is a mix of Dirichlet and Robin conditions. A recent paper [48] has proven existence and the comparison principle for the Monge-Ampère with oblique boundary conditions in higher dimensions. This is enough motivation for monotone Robin discretizations.

5.1.5 Implementation Details

We implemented the two DDMs in MATLAB using its nonlinear solver and parallel computing tools. The local equations $F_i^h = 0$ and $F_{i,o}^h = 0$ are solved with the trust-region algorithm in `fsolve`. The local solves use `parfor` on an eight-core personal computer, allowing up to eight concurrent subdomains. Boundary updates and residual calculations run serially on the main thread.

There is no guarantee of convergence at this stage, so the DDM was programmed to return one of several numerical codes depending on circumstance. The loop runs at most N_{max} number of times. The loop can exit earlier if the residual falls below the given tolerance, i.e. $r_n < \epsilon$ in which case the DDM returns code 0 to signify convergence. An alternate form of convergence is when the condition

$$\frac{|r_n - r_{n-1}|}{r_n} < \epsilon_c$$

is satisfied a consecutive number of times. In this case, the return code is 1. Code 1 means the cost of performing an additional iteration is no longer worth the reduction in error. There were some cases in the experiment where the residual decreased then leveled out to a value above the tolerance. Code -1 signifies divergence. This occurs when the residual grows a consecutive number of times. If the iteration count hits N_{max} then the return code is -2. Code -2 could mean that the residual oscillates or decays slowly. To summarize:

- code 0 means convergence,
- code 1 means alternative convergence,
- code -1 means divergence,
- code -2 means inconclusive.

All of the stopping parameters, the $N_{max}, \epsilon, \epsilon_c$, were chosen liberally to capture the true nature of the DDM and were changed numerous times to avoid misdiagnosis.

To make the comparison between the NDDM and the ODDM fair, we choose the same N for both regimes. The meshes for both methods are uniformly spaced grids with $h = (b-a)/(N+1)$. The ODDM benefits from having a uniform mesh because it simplifies the transfer of the solution to the neighboring subdomain. In the NDDM, on the other hand, the uniform mesh is unnecessary. It can have different h_i for each subdomain $i = 1, \ldots, N_d$ and it would not change the difficulty of or the computational complexity of the method. The ODDM will have an overlap parameter of the form $\delta = 2mh$, where m is a positive integer. The integer m is effectively the overlap parameter which can be inferred as the number of nodes the subdomains extend over to the neighboring subdomain.

5.1.6 Results for the Non-overlapping Method

We first examine the transmission parameter β. Table 5.1 reports selected values for case 1 with $N_d = 2$ and $N = 200$. A nonzero imaginary part produces oscillating

residuals, while negative real values lead to divergence, possibly because monotonicity is lost. The apparent convergence at $\beta = 0$ does not recover the desired coupling and is treated as a failure.

Table 5.1 Performance of the NDDM for Various β

β	Return Code	Iterations	Runtime (s)	Error
-1.00	-1	5	1.835	1.088e+00
i	-2	300	41.145	8.675e-01
0.00	0	1	0.130	6.544e-01
0.01	0	162	6.499	3.355e-01
1.00	0	17	3.629	1.058e-02
1.50	0	13	2.154	1.058e-02
2.00	0	12	2.018	1.058e-02
2.50	0	9	1.349	1.058e-02
3.00	0	6	0.861	1.058e-02
3.50	0	10	1.514	1.058e-02
4.00	0	12	1.791	1.058e-02

In these experiments, useful convergence occurs for real $\beta > 0$. The third column gives the iteration count. The results suggest a minimum near $\beta_{op} \approx 3.0$ in Table 5.1. The optimal value is sensitive to the discretization, decomposition, and PDE. With other factors fixed, we conjecture a convex dependence of the iteration count on β over $(0, \infty)$.

Figure 5.6 explores this dependence for $\beta \in (2, 4)$ in increments of $\Delta\beta = 0.1$. The left plot uses $N_d = 2$ and the right $N_d = 4$. The preferred value changes with N_d and changes again when $N = 400$. Thus the optimal β_{op} depends on the decomposition and resolution. Across the three regularity cases, the optimal values are similar; the endpoint singularity does not noticeably shift them in these tests.

Table 5.2 reports iteration counts and wall-clock times as N_d varies, using $N = 200$ and fixed $\beta = 3$. Counts increase with the number of subdomains, particularly in case 3. This suggests that the boundary discontinuity at $x = 1$ can impair convergence when the transmission parameter is not retuned.

In the follow up experiment, the transmission parameter was numerically optimized. A parameter sweep was performed to find the optimal β for case 1.

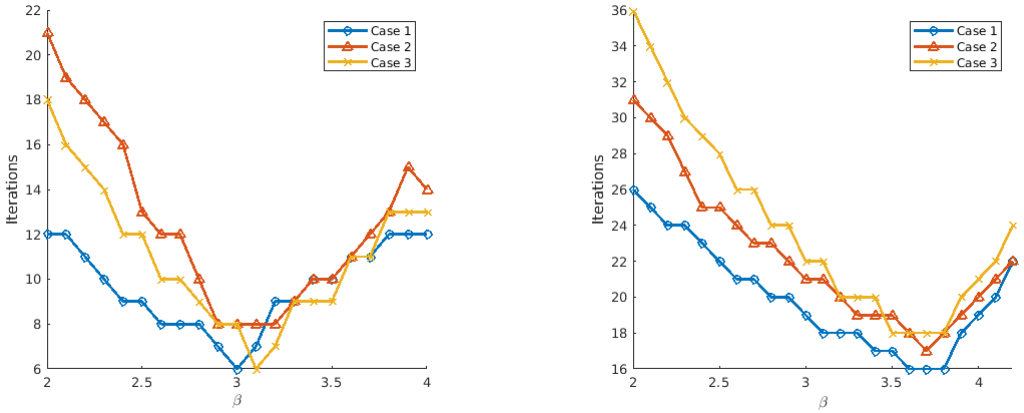


Figure 5.6 Iteration numbers are plotted with respect to β for two subdomains (left) and four subdomains (right).

Table 5.2 Performance of the Non-optimized NDDM

	Case 1		Case 2		Case 3	
N_d	Iter	Runtime	Iter	Runtime	Iter	Runtime
2	4	1.3	6	2.0	5	1.7
3	8	1.2	15	3.2	18	3.0
4	11	1.4	26	4.0	33	4.3
5	18	1.9	29	3.8	71	7.8
6	26	1.9	43	4.5	261	23.3
7	33	2.2	41	4.0	628	48.5
8	40	2.5	48	4.1	1369	96.9

The β_{op} computed for Case 1 was used for Cases 2 and 3. Table 5.3 shows the result of the same experiment except with optimal transmission operators. Case 3 exhibited a linear increase in iteration very similar to cases 1 and 2. This suggests that the NDDM's convergence is independent of the regularity given that the optimal transmission parameter is used.

5.1.7 Results for the Overlapping Method

Next, we discuss the numerical results for the ODDM. We examine how the overlap $\delta = 2mh$ affects convergence across the three regularity cases. Increasing overlap enlarges the local problems and duplicates work, an effect that becomes more costly

Table 5.3 Performance of the NDDM with Optimized β

		Case 1		Case 2		Case 3	
N_d	β_{op}	Iter	Runtime	Iter	Runtime	Iter	Runtime
2	3.0	4	1.63	6	1.42	5	1.50
3	3.3	10	1.47	9	1.54	11	1.42
4	3.6	16	1.74	19	1.92	18	1.60
5	4.0	20	1.96	22	2.07	23	1.79
6	4.1	25	2.07	25	2.22	26	1.91
7	4.3	30	2.27	34	2.50	35	2.26
8	4.4	35	2.62	40	2.76	41	2.66

in higher dimensions. We also compare stopping criteria based on changes in Dirichlet data and changes over the entire overlap.

Table 5.4 compares these residual definitions with $N = 200$, $N_d = 2$, and varying m. The overlap-based criterion requires roughly 50 percent more iterations than the Dirichlet-data criterion. Larger overlaps accelerate convergence. Direct comparison is complicated by differences in final error and stopping parameters. Changing the overlap residual from the one-norm to the two-norm or infinity-norm has little effect, so subsequent experiments use the Dirichlet-data residual. Iteration counts increase slightly from case 1 to cases 2 and 3.

Table 5.4 Comparison of the Two Residual Definitions

	Case 1		Case 2		Case 3	
m	Dirich.	Overlap	Dirich.	Overlap	Dirich.	Overlap
1	128	179	135	187	139	193
2	92	135	96	142	99	146
3	72	110	76	115	78	119
4	60	93	63	98	65	101
5	52	81	54	85	56	88
10	32	51	34	54	34	55
15	24	38	24	40	26	41
20	19	31	20	32	20	33

Table 5.5 reports ODDM performance as the number of subdomains varies, with $N = 200$ and $m = 5$. Iteration counts grow approximately linearly with N_d, with modest differences among the regularity cases. These differences are small at $N_d = 2$ and more visible at $N_d = 8$. Runtime also increases with N_d.

Table 5.5 Performance of the 1D ODDM

	Case 1		Case 2		Case 3	
N_d	Iter	Runtime	Iter	Runtime	Iter	Runtime
2	52	1.686	54	1.702	56	1.743
3	81	2.813	86	3.015	89	3.081
4	110	3.320	117	3.493	121	3.631
5	139	4.715	148	4.957	153	5.091
6	169	5.684	180	5.983	186	6.071
7	199	7.536	212	7.913	220	8.032
8	227	7.861	242	8.482	252	8.617

5.1.8 Concluding Remarks

We adapted overlapping restricted additive Schwarz (ODDM) and non-overlapping optimized Schwarz (NDDM) to the semilinear 1D Monge-Ampère equation using a monotone finite difference discretization. Both methods converge in the three regularity cases tested.

NDDM imposes Robin interface conditions, which require derivative information that may not exist for a general viscosity solution. In these experiments, real positive β produces convergence. Tuning β reduces iteration counts and runtime below those of ODDM. It also mitigates the effect of solution regularity. The observed parameter dependence suggests that numerical optimization is practical.

ODDM uses local Dirichlet problems and does not impose interface differentiability. Its performance is more consistent across the three regularity cases. Overlap can be tuned to reduce runtime, although our implementation did not achieve a parallel speedup.

Increasing the number of subdomains increased runtime in this MATLAB implementation. Parallel communication and implementation overhead may have outweighed the savings from smaller local solves. More careful tuning of `fsolve`, including limiting local function evaluations, may improve performance. These experiments establish numerical convergence but motivate a more efficient implementation before assessing parallel scalability.

5.2 Domain Decomposition Methods for the Two-dimensional Monge-Ampère Equation

For the 2D equation $\det(D^2u(x)) = f(x)$ on a rectangular domain, we focus on overlapping restricted additive Schwarz methods. Robin conditions in optimized Schwarz may impose additional regularity that is not supported by the viscosity framework used here.

To prioritize computational efficiency, we implemented the 2D methods in C using the Portable, Extensible Toolkit for Scientific Computation (PETSc) [5]. PETSc provides efficient linear algebra and communication routines for nonlinear domain decomposition. We compare Newton-Krylov-Schwarz (NKS) and Schwarz-Newton-Krylov (SNK) across a range of Monge-Ampère test problems.

5.2.1 Two-dimensional Model Problem

We are interested in the problems of the form

$$\begin{cases} \det(D^2u(x)) = f(x), & x \in \Omega \\ u(x) = g(x), & x \in \partial\Omega \\ u \text{ is convex.} \end{cases} \tag{5.14}$$

where $\Omega \subset \mathbb{R}^2$ is a rectangular domain i.e. $\Omega = [x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}]$. We require $f(x)$ to be nonnegative and $g(x)$ to be continuous. We consider four representative examples of the Monge-Ampère equation:

1. Smooth solution $u \in C^\infty(\overline{\Omega})$ where $\Omega = (-1, 1)^2$

$$u(\mathbf{x}) = \exp\left(\frac{\|\mathbf{x}\|^2}{2}\right), \quad f(\mathbf{x}) = \left(1 + \|\mathbf{x}\|^2\right)\exp\big(\|\mathbf{x}\|^2\big). \tag{5.15}$$

2. Fully degenerate solution $u \in C^1(\Omega)$ where $\Omega = (0, 1)^2$ and $\mathbf{x}_0 = (0.5, 0.5)$

$$u(\mathbf{x}) = \frac{1}{2}\big((\|\mathbf{x} - \mathbf{x}_0\| - 0.2)^+\big)^2, \quad f(\mathbf{x}) = \left(1 - \frac{0.2}{\|\mathbf{x} - \mathbf{x}_0\|}\right)^+. \tag{5.16}$$

Fully degenerate means $D^2u(x) = 0$ for some x in Ω. The superscript + symbol means $(f)^+ := \max(f, 0)$.

3. The hemisphere on $\Omega = (-1, 1)^2$ with a gradient singularity at the corner $(1, 1)$

$$u(\mathbf{x}) = -\sqrt{2 - \|\mathbf{x}\|^2}, \quad f(\mathbf{x}) = 2\big(2 - \|\mathbf{x}\|^2\big)^{-2}. \tag{5.17}$$

4. Semi-degenerate solution $u \in C^2(\Omega)$ where $\Omega = (0, 1)^2$ and $\vec{\gamma} = (0.6, 0.4)$

$$u(\mathbf{x}) = (\vec{\gamma} \cdot \mathbf{x})^2, \quad f(\mathbf{x}) = 0. \tag{5.18}$$

Semi-degenerate means D^2u has one zero and one nonzero eigenvalue.

These are the examples considered in [22]. The varying levels of solution regularity and eigenvalue properties are expected to have an impact on the performance of the numerical solver.

5.2.2 Problem Parameters

This section describes the domain, discretization, solver settings, and computational resources used in the experiments.

We may choose the limits of our rectangular domain. The four parameters in $\Omega = [x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}]$ are free. In general, we may discretize the interior of the domain with a lattice of N_x by N_y uniformly spaced mesh nodes. Since the domain of our examples are square, we mainly use $N_x = N_y = N$. Thus, the step size for the grid is usually

$$h = \frac{x_{max} - x_{min}}{N + 1} = \frac{y_{max} - y_{min}}{N + 1}. \tag{5.19}$$

The $N_x \times N_y$ lattice of nodes are divided into subgroups in an m by n block format. These blocks correspond to the domain decomposition subdomains. We may either choose m and n directly, or we choose the number of subdomains N_d and let PETSc automatically decompose the domain.

The PDE in Equation (5.14) is discretized using the determinant integral representation quadrature discussed in Subsection 4.2.3. Simpson's rule method is

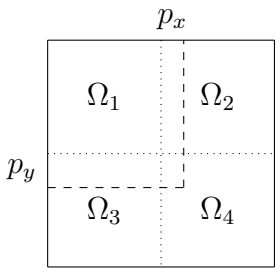


Figure 5.7 Overlapping decomposition with $N_d = 4$ subdomains

used for the quadrature. The integer stencil width is chosen to be $d = \lceil h^{-1/3} \rceil$. The regularizing parameter is $\epsilon = (hd)^2$. The overlap between the subdomains are free to control. The two integers δ_x and δ_y, which determine how many layers of nodes the subdomains should extend in the x and y directions. We will discuss the overlap amount in terms of percentages: x-overlap percentage $p_x := \frac{\delta_x}{N_x/n}$ and y-overlap percentage $p_y := \frac{\delta_y}{N_y/m}$. Figure 5.7 illustrates an example with the subdomain Ω_1 enlarged with $p_x = 20\%$ and $p_y = 30\%$. Note that 100% overlap is the fully redundant case where each subdomain solves the full problem. In this work, we only consider uniform x and y overlap $p_x = p_y =: p$.

All simulations were run on NJIT's Stheno cluster. The available specifications listed 32 nodes, 3840 GB of RAM, and 392 cores. Jobs used at most nine nodes, with one subdomain per node, so the largest decomposition was a 3×3 block configuration. The jobs allocated approximately 12 GB of memory per subdomain. For comparison, a dense 40000×40000 double-precision matrix requires approximately 11.92 GB. We requested one task per node to avoid sharing the allocated memory between subdomains.

There are three main components of the PETSc solve routine that can be customized: the nonlinear solver, the linear solver, and the linear preconditioner. For the nonlinear solver we consider Newton's method `newtonls` and the nonlinear additive Schwarz `nasm`. For the linear solver, we mainly only consider the deflated GMRES `dgmres` [37] or the pipelined GMRES `pfgmres`. For the preconditioners, the

main choices are the Eisenstat SSOR `eisenstat` [35], incomplete LU `ilu`, and the additive Schwarz `asm`. For certain nonlinear solvers, such as Newton's method, there is also a line search method can be selected as well. We use the cubic backtracking `bt`. The combination of choices correspond to different DDMs. They will be discussed in more detail in a later section. Other available methods can be found in the user manual [4].

We have control over the stopping condition of the iterative nonlinear solver. The safest choice is to terminate when the relative residual falls below a certain tolerance. The default tolerance is 10^{-8}. We may also configure the solver to stop at an absolute residual falls below a tolerance. The only issue with the absolute residual is each problem has a different initial residual and it may be difficult to find a robust tolerance level. It is also possible to set a maximum number of iterations. Alternatively, we may also stop after a set amount of evaluations of the residual function $\mathbf{r}(\mathbf{u}) = \mathbf{F}(\mathbf{u}) - \mathbf{b}$. If an iterative method is chosen for the linear solver, such as GMRES, then the stopping conditions for it can be changed in the same way. The default relative residual tolerance for the GMRES is 10^{-5}.

5.2.3 Serial Algorithm

Before moving on to the DDMs, we discuss the serial version of the algorithm which is Newton-Krylov method with line search. We use deflated restarting GMRES for the Krylov method, Eisenstat type symmetric SOR for the preconditioner, and cubic backtracking for the step size selection. The deflated GMRES is necessary for $N \geq 400$ because the standard restarting GMRES would fail to converge. The Eisenstat preconditioner was chosen out of several options because it yielded the fastest computation time. Figure 5.8 shows the computational runtime of the serial algorithm for all four examples with respect to N. The relative residual tolerance for the Newton and Krylov methods are set to 10^{-8} and 10^{-5}, respectively.

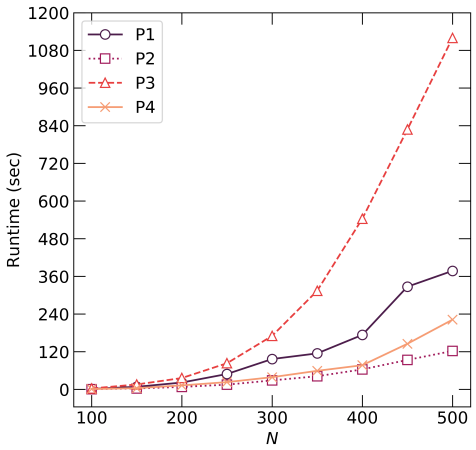


Figure 5.8 Runtime comparison for the four examples on a $N \times N$ mesh.

Figure 5.8 shows the runtime of the serial algorithm for each of the four examples on an $N \times N$ mesh. The runtimes for the examples are initially comparable, but as the mesh size N, grows the difficulty of the individual examples become more pronounced. Example 3 with the derivative singularity is the most difficult problem. At $N = 500$, the runtime of example 3 is more than three times larger than that of example 1. Although examples 2 and 4 have solutions with unfavorable Hessian properties, their runtimes are small compared to the runtime of example 1 which has a smooth solution. The result suggests that the problem size handled by a single processor should not exceed 200×200. Otherwise, there would be large deviations in the performance across the four examples.

5.2.4 NKS and SNK

In **Newton-Krylov-Schwarz (NKS)**, Newton's method with a backtracking line search is the outer solver. Pipelined GMRES solves each Jacobian system with right additive Schwarz preconditioning. The preconditioner's local systems are approximated with ILU, and their overlap can be varied.

The second type of DDM is the Schwarz-Newton-Krylov (SNK). The SNK uses the NASM as the outer solver and Newton-Krylov for the local nonlinear problems.

The local solver uses Newton's method with cubic backtracking line search. The linear solve is handled by the deflated GMRES left-preconditioned with the Eisenstat SOR. The NASM combines the local solutions with restriction up to the non-overlapping boundary. This method can be thought of as the additive Schwarz method adapted to the nonlinear problem. The only difference is instead of defining the residual based on the change in Dirichlet data, it is based on nonlinear residual $\mathbf{r}_i(\mathbf{u}_i) = \mathbf{F}(\mathbf{u}_i) - \mathbf{b}$. When the relative residual $\|\mathbf{r}_i\| \,/\, \|\mathbf{r}_0\|$ falls below a tolerance, the method is said to converge.

PETSc also supports the nonlinear compositions discussed in Subsection 4.3.4. We tested NGMRES, FAS, ASPIN, and RASPEN with the wide stencil Monge-Ampère discretization, but did not obtain runtimes better than serial Newton-Krylov. SNK and NKS performed better with less tuning. Those additional experiments are omitted. More detailed algebraic analysis and solver tuning may help assess their potential.

5.2.5 Numerical Results

In this section, we present a series of numerical experiments for the NKS and SNK for the 2D Dirichlet Monge-Ampère problem. They are broken into four sections: under-solving, subdomain overlap, domain size, and scaling.

Many tests use $N_d = 4$ and $N = 200$. Table 5.6 gives the serial baseline at $N = 200$, with relative tolerances 10^{-8} for Newton iteration and 10^{-5} for deflated GMRES. Krylov counts are summed over all Newton steps. This baseline supports comparisons of speedup and error.

Under-solving the Local Problem Accurate local solves are not always efficient. Initial SNK tests were slower than the serial method because too much time was spent in local Newton iterations. Relaxing local solve accuracy was essential to obtaining a speedup. A similar adjustment can also improve NKS.

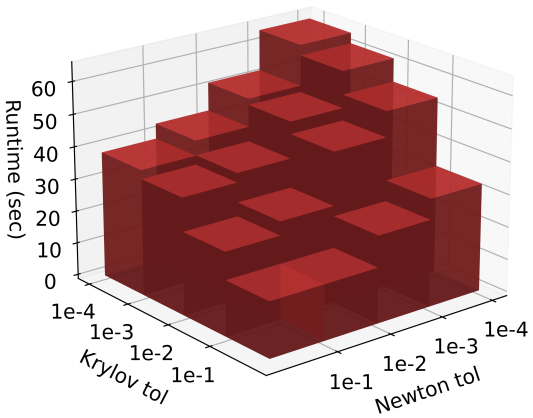
Table 5.6 Stats for the 2D Serial Algorithm with $N = 200$

	Ex. 1	Ex. 2	Ex. 3	Ex. 4
Time (s)	21.743	7.609	35.814	12.555
Error	5.259e-04	8.340e-05	5.955e-05	1.081e-04
Newton Iter.	32	13	45	16
Krylov Iter.	1269	310	1529	778

NASM local solves use incomplete global information. In the decompositions considered here, information takes on the order of $\sqrt{N_d}$ iterations to reach all subdomains. Relaxing local Newton or Krylov tolerances reduces the work between exchanges. This can avoid expensive solves with inaccurate neighboring data, but may increase the outer iteration count; overly aggressive under-solving can cause divergence. Runtime optimization therefore balances local computation against communication and depends on the hardware.

Figure 5.9 shows the SNK runtime for example 3 with relaxed residual tolerances for both the Newton and Krylov methods. The tolerances are relaxed in powers of 10 starting from 10^{-4} down to 10^{-1}. The parameters of the problems are $N = 200$, $N_d = 4$, and 10% overlap between the subdomains. Note that, since the global tolerance is fixed at 10^{-8}, the final errors are all on the same order regardless of the inner tolerances.

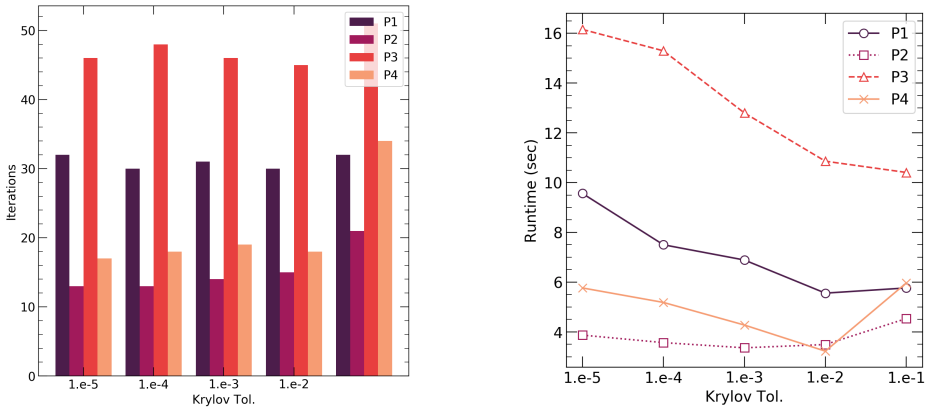
Relaxing both tolerances from 10^{-4} to 10^{-1} reduces the runtime by more than a factor of six for example 3. Examples 1, 2, and 4 improve by approximately factors of four, five, and eight. The tested runtimes decrease monotonically as the tolerances are relaxed. Further experiments found that a single Newton step with a second-order backtracking line search can perform well. We call this variant single-iteration Newton (SIN). A Krylov tolerance near 10^{-1} is generally effective, although tuning may be needed. SIN and HTN results are compared in Subsection 5.2.5.



Newton \ Krylov	1e-4	1e-3	1e-2	1e-1
1e-4	49	49	49	50
1e-3	49	49	49	50
1e-2	49	49	49	49
1e-1	51	49	49	70

Figure 5.9 Runtime (left) and iteration number (right) for Example 3 with respect to relaxed Krylov and Newton tolerances.

Next, we discuss the under-solving for the NKS. We, again, use the parameters $N_d = 4$ and $N = 200$ for all four examples. We use zero overlap for the additive Schwarz. We relax the Krylov tolerance by orders of magnitude, tol $= 10^{-5}$, 10^{-4}, 10^{-3}, 10^{-2}, and 10^{-1}. The computation results are shown in Figure 5.10.



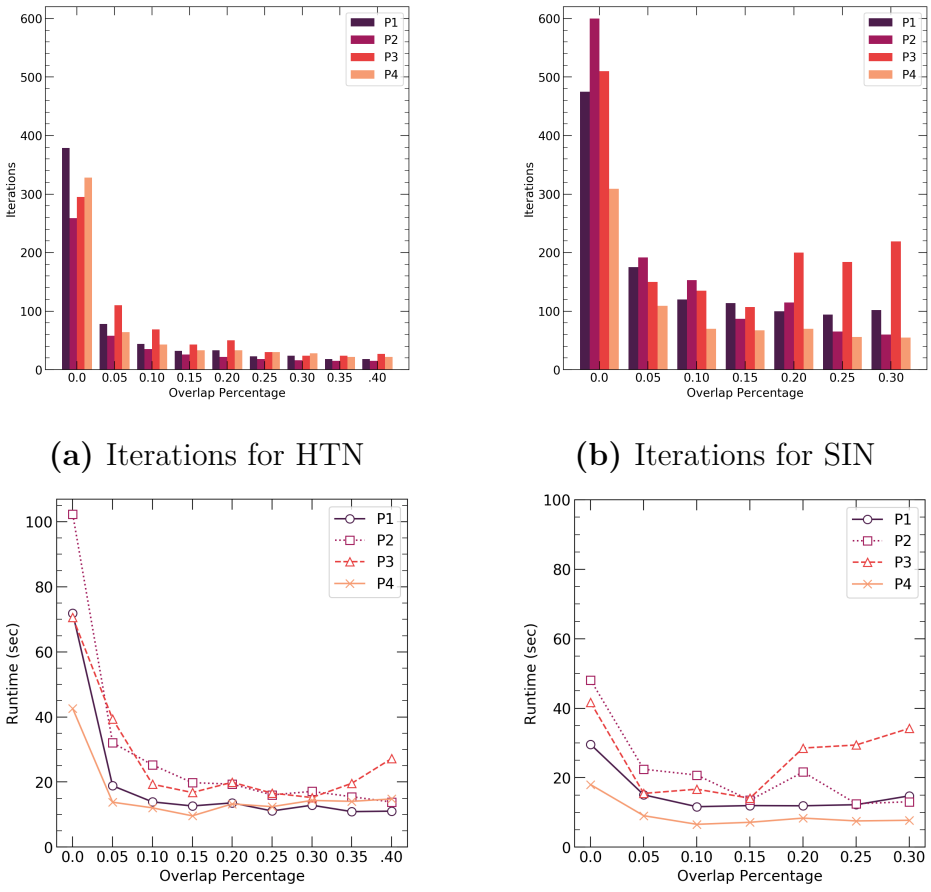
(a) Newton iterations in the NKS **(b)** Runtime of the NKS

Figure 5.10 The plots show how under-solving affects the performance of the NKS.

The left plot gives Newton iteration counts, which generally increase as the Krylov tolerance is relaxed. Example 4 is the most sensitive. The right plot gives

wall-clock times: examples 1 and 3 improve monotonically, while examples 2 and 4 are fastest for tolerances between 10^{-3} and 10^{-2}. NKS performs well at the default 10^{-5}, but tuning the Krylov tolerance can improve its runtime.

Overlap Overlap is a principal tuning parameter for additive Schwarz. More overlap often reduces iteration counts but increases local solve costs. The best choice depends on the problem, discretization, and hardware. We examine this tradeoff numerically.



(a) Iterations for HTN

(b) Iterations for SIN

(c) Runtime for HTN

(d) Runtime for SIN

Figure 5.11 Comparison of the HTN and SIN with respect to subdomain overlap.

First, the two SNK algorithms, the HTN and SIN are tested. The overlap tests are performed on a 200×200 mesh split into $N_d = 4$ subdomains. Overlaps range

from 0 to 50% in increments of 5% were tested. Figure 5.11 shows the iteration number and the runtime for the SNK algorithms. The iteration number decreases with respect to increase in overlap. The only exceptions are examples 2 and 3 for the SIN. The plot of the runtimes show similar trend. This experiment was repeated for $N = 100$ and $N = 300$ (not shown here), but the optimal overlap was hardly affected at all. This suggests that the actual size of the local system is not the main factor, but rather the relative size of the local system compared to global size.

For the SNK, choosing zero overlap between the subdomains is ill advised. The steepest change in iterations occurs between 0% and 5% overlap for all examples. Next, the SIN and HTN can achieve the same level of runtime when they are tuned. In Figure 5.11, the HTN with 25% performs just as well as the SIN with 5% overlap.

We also test the overlap for the NKS. The overlap for the NKS is controlled by the additive Schwarz preconditioner. Greater overlap for the preconditioner may result in lower iteration number for the Krylov solver. However, the size of the local sub-blocks will increase. The results of the NKS runtime for $N = 200$ and $N_d = 4$ is shown in Figure 5.12.

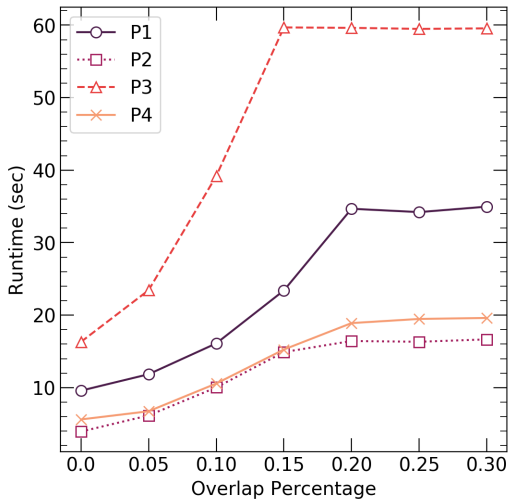


Figure 5.12 Performance of the NKS with respect to overlap.

For the tested NKS configurations, increasing overlap increases runtime, and zero overlap is best. At 25% overlap, total Krylov counts are 677, 143, 841, and 321 for the four examples, approximately half the serial counts in Table 5.6. Thus fewer Krylov iterations do not guarantee a faster solve: the larger local ILU systems outweigh the iteration savings.

Domain Size A series of tests were performed to see the effect of varying the domain size on the performance of the DDMs. We are primarily concerned with the error $\left\|u - \mathbf{u^h}\right\|_2$ is affected by the size of the problem. The symmetric example $u(\mathbf{x}) = \exp\left(\frac{\|\mathbf{x}\|^2}{2}\right)$ on a square domain $[-L, L]^2$, for L =0.5, 1.0, 1.5, or 2.0 is the primary test case. The idea is to fix h while making the domain larger. We tested $h = 0.05$ and $h = 0.01$. The size of the discrete problem on length scale L with step size h is on the order of $(2L/h)^2$.

For the first set of tests, we wished to see effect of domain size on the local solver. The SNK uses Newton-Krylov as the local solver which is controlled by the Newton tolerance and the GMRES tolerance. Table 5.7 shows the final error for the serial algorithm with respect to tolerance, domain size, and step size. The rows of the table show the relative residual tolerance of both the Newton's method and the GMRES. For example, the errors shown in the top row gives us an idea of the quality of local solutions computed in each iteration of the HTN. The main result from this experiment is that increasing the domain, for a fixed h, will increase the error for example 1. This illustrates another computational challenge for the Monge-Ampère equation.

In the next set of tests, we were concerned with comparing the DDMs' performance with respect to domain size. We let the outer nonlinear solver iterate until the relative residual falls below a factor of 10^{-8}. First, we show the result for the serial Newton-Krylov algorithm in Table 5.8. The runtime grows rapidly with

Table 5.7 Final Error of the NK Solver with respect to L and Tolerance

	$L = 0.5$		$L = 1.0$		$L = 1.5$		$L = 2.0$	
Tol.	$h = 0.05$	$h = 0.01$	$h = 0.05$	$h = 0.01$	$h = 0.05$	$h = 0.01$	$h = 0.05$	$h = 0.01$
10^{-1}	7.627e-1	1.077e0	1.548e0	1.495e0	5.048e0	3.172e0	1.142e2	9.129e1
10^{-2}	4.643e-2	4.963e-1	2.877e-1	1.681e0	2.248e0	8.640e0	4.041e1	2.400e1
10^{-3}	6.464e-3	6.058e-2	6.755e-2	8.842e-1	2.200e-1	5.836e0	3.838e0	7.149e1
10^{-4}	1.214e-3	9.970e-4	5.740e-3	1.562e-1	3.501e-2	1.181e-1	3.298e-1	4.730e0
10^{-5}	1.126e-3	1.937e-4	4.630e-3	1.426e-2	1.973e-2	4.049e-2	1.290e-1	2.528e-1
10^{-6}	1.127e-3	1.395e-4	4.624e-3	9.333e-4	1.723e-2	2.390e-3	1.224e-1	2.483e-2

respect to L for $h = 0.01$. The results for the NKS with $N_d = 4$ subdomains and zero overlap are shown in Table 5.9. The NKS notably has a factor of 4 speedup on the problems of larger sizes on $h = 0.01$. The error of the NKS is almost unchanged compared to the NK results.

Table 5.8 Results for NK on Example 1 with Tolerance 10^{-8}

L	Iters	Error	Runtime
0.5	10	1.127e-03	0.0127
1.0	14	4.638e-03	0.0863
1.5	33	1.723e-02	0.5925
2.0	86	1.221e-01	2.9904

(a) $h = 0.05$

L	Iters	Error	Runtime
0.5	13	1.283e-04	0.8710
1.0	31	5.258e-04	25.5889
1.5	74	1.973e-03	193.9611
2.0	386	1.409e-02	2212.7946

(b) $h = 0.01$

Table 5.9 Results for NKS on Example 1 on 4 Subdomains

L	Iters	Error	Runtime
0.5	10	1.127e-03	0.0475
1.0	13	4.638e-03	0.0950
1.5	35	1.723e-02	0.4472
2.0	111	1.221e-01	2.4075

(a) $h = 0.05$

L	Iters	Error	Runtime
0.5	13	1.283e-04	0.5725
1.0	32	5.258e-04	8.2575
1.5	55	1.974e-03	43.6846
2.0	274	1.410e-02	474.1449

(b) $h = 0.01$

The results for the SIN are shown in Table 5.10. The main tuning parameters of the SIN are the overlap percentage and the Krylov tolerance. For each L and h, a range of values for the overlap and Krylov tolerances have been tested.

Table 5.10 shows the best performance of the SIN out of the overlap percentages $5\%, 10\%, \ldots, 40\%$ and Krylov tolerances $10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}$. For the cases $L = 1.5, 2.0$ and $h = 0.01$, the SIN failed to converge within a reasonable amount of time for any of the parameter values. For the cases where the SIN converged, the error of the solution was lower than the NK and NKS. Restricting the number of Newton iterations may be optimal in some cases, but in other cases the SNK will ultimately fail.

Table 5.10 Results for SIN on Example 1 on Four subdomains

L	Iters	Error	Runtime
0.5	16	1.123e-03	0.020778
1.0	21	4.626e-03	0.080246
1.5	92	1.707e-02	0.554163
2.0	444	1.209e-01	4.611878

(a) $h = 0.05$

L	Iters	Error	Runtime
0.5	63	3.623e-05	1.183388
1.0	31	1.278e-04	11.922407
1.5	NA	NA	NA
2.0	NA	NA	NA

(b) $h = 0.01$

Table 5.11 Results for HTN on Example 1 on 4 Subdomains

L	Iters	Error	Runtime
0.5	12	1.126e-03	0.021420
1.0	12	4.629e-03	0.076236
1.5	19	1.713e-03	0.379458
2.0	9	1.210e-01	1.389934

(a) $h = 0.05$

L	Iters	Error	Runtime
0.5	10	3.159e-05	0.636704
1.0	15	1.684e-04	8.520799
1.5	16	3.152e-03	62.698950
2.0	43	9.748e-02	1099.567024

(b) $h = 0.01$

Table 5.11 reports the fastest HTN runs obtained by tuning Newton and Krylov tolerances and overlap. The preferred Newton tolerance is 10^{-1} in most cases, while the Krylov tolerance is 10^{-1} or 10^{-2}. The optimal overlap varies substantially. HTN is faster than SIN except for $L = 0.5$ with $h = 0.05$. For the two largest cases, $L = 1.5, 2.0$ with $h = 0.01$, SNK converges with Newton tolerance 10^{-1} but is slower than NKS.

Scaling In this section we discuss the strong and weak scaling of the DDMs. We compare the SIN, HTN, and the NKS for decompositions of $N_d = 2, 4, 6, 8, 9$. There are two common benchmarks for parallel algorithms: the strong scaling and the weak scaling. Let us denote the serial computational runtimes in Figure 5.8 by T_1. The wall-clock runtime of the DDM version of the algorithm with p workers is denoted by T_p. The **speedup** is defined to be the ratio

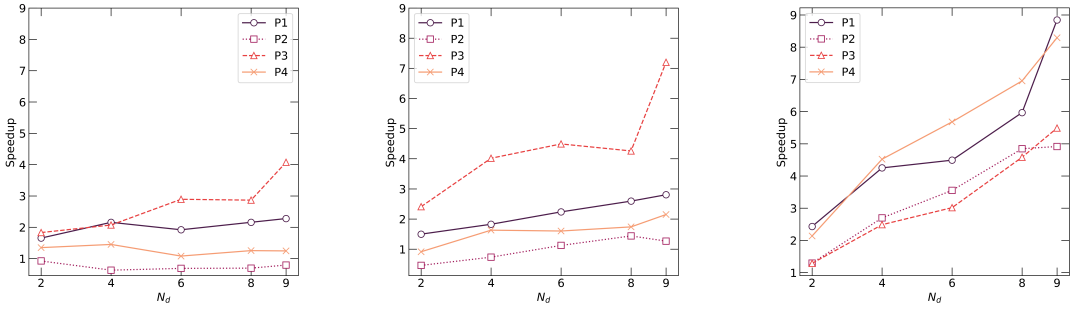
$$S = \frac{T_1}{T_p}. \tag{5.20}$$

In the ideal case, employing p workers should yield a speedup of around p. There are cases in which the speedup can surpass the ideal amount. When speedup is greater than p, this is called **superlinear** speedup. The **parallel efficiency** of a DDM algorithm is defined to be

$$E = \frac{S}{p}. \tag{5.21}$$

The ideal efficiency is $E = 100\%$, corresponding to perfectly balanced work and no communication overhead. **Strong scaling** measures speedup as p increases for a fixed problem size N. **Weak scaling** measures efficiency as the problem size grows with the number of workers while the workload per worker remains fixed.

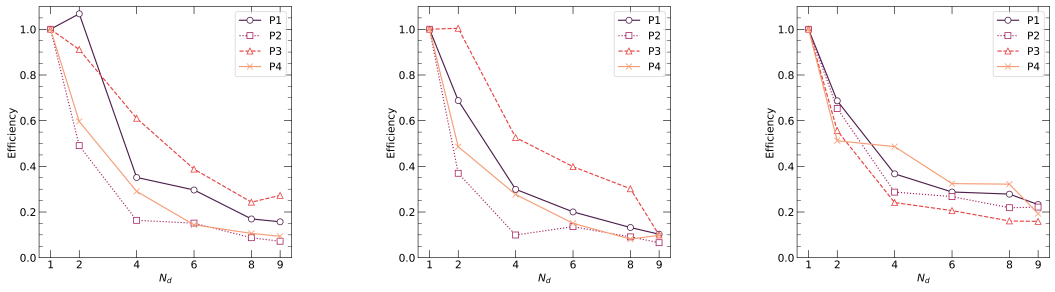
For the strong scaling, the mesh size is fixed to 300×300. The result of the strong scaling test is plotted in Figure 5.13. The speedup is computed with respect to the serial runtime shown in Figure 5.8. The HTN was run with overlap $p = 0.3$, the SIN with $p = 0.1$, and the NKS with no overlap. The ideal strong scaling plot should be a linear trend with slope 1. The closest to the ideal scaling is seen in the NKS for examples 1 and 4. The scaling for example 3 is notably worse on the NKS. The SIN shows better speedup for example 3 than the NKS. The other examples, however, are worse than the NKS. The HTN does not show good scaling for any of the examples.



(a) Speedup for HTN **(b)** Speedup for SIN **(c)** Speedup for NKS

Figure 5.13 Strong scaling test with $N = 300$.

For the weak scaling test, the local mesh size of the subdomain is fixed to 150. The overlap percentages are fixed. Depending on N_d, the global $N_x \times N_y$ mesh is adjusted so that workload of the local solver is always the same. Figure 5.14 shows the results of the weak scaling test for the three DDMs. It is normal to see efficiency drop as N_d increases due to the increase in inter-node communication. The plot, however, shows a sharp drop-off in efficiency for all methods. The notable exceptions are the SNK methods for $N_d = 2$. They have high efficiency for examples 1 and 3. Aside from these examples, the DDMs show poor weak scaling.



(a) HTN **(b)** SIN **(c)** NKS

Figure 5.14 Weak scaling test.

Neither scaling test gives uniformly strong results. The overlap was not retuned for these experiments, and using the same mesh size $N \times N$ across examples does not equalize their computational difficulty. Each example may require a different local

workload for efficient parallel execution. Further experiments could tune both the local mesh size and overlap for each example.

Example 5 We consider another case for the Monge-Ampère equation which we will refer to as example 5:

$$u(\mathbf{x}) = \max\left(\|\mathbf{x}\|_2 - \frac{1}{5}, 0 \right)^{5/2}, \quad f(\mathbf{x}) = \frac{3}{8}\max\left(5\|\mathbf{x}\|_2 - 1, 0 \right)^2. \tag{5.22}$$

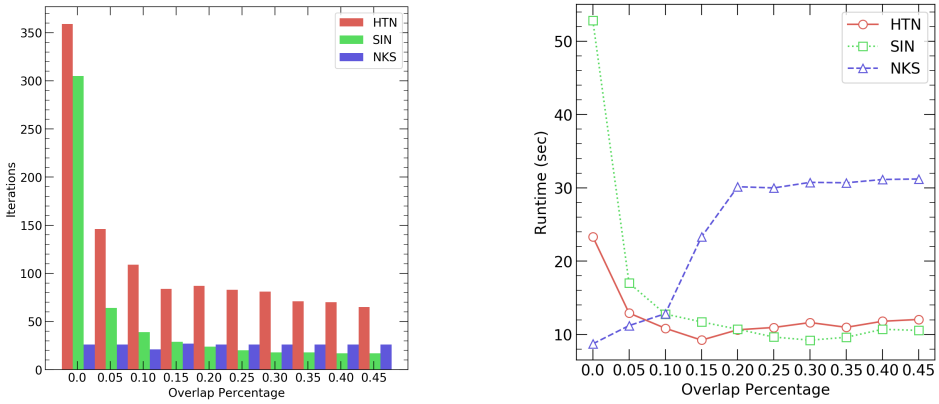
This solution features a radially symmetric parabolic shape with a flat circular bottom in the region $\{\|\mathbf{x}\|_2 < 1/5\}$ which corresponds to degeneracy. Also, the gradient of the solution gets larger as $\mathbf{x}$ moves farther from the origin similar to example 1. Table 5.12 shows the runtime of the serial NK algorithm on an $N \times N$ mesh size for several N. Compared to the serial runtime of the other examples shown in Figure 5.8, example 5 is most similar to example 1.

N	100	150	200	250	300	350	400	450
Runtime (s)	2.153	9.658	27.942	66.941	110.184	206.355	295.405	445.789

Table 5.12 Runtime of the NK method for example 5 with respect to mesh size.

We test the performance of DDMs for example 5 on a 200×200 mesh with $N_d = 4$ subdomains configured in a 2×2 block format. First, the subdomain overlap is varied in terms relative coverage of the subdomain. The iteration number and wall-clock runtime are shown plotted in Figure 5.15. The figure shows the results of the HTN, SIN, and the NKS. For the HTN and SIN, the iteration number monotonically decreases with respect to overlap percentage. As for the runtime, the HTN is minimized at 15% and the SIN at 30%. The overlap percentage has a relatively low impact on the NKS iteration number (Newton iterations). As for the runtime, the NKS performs best with no overlap. When the overlap is optimized, each of the DDMs will roughly achieve a speedup ratio around 3. The results for example 5 are

within expectation. They are similar to the results of the overlap test for the other examples.



(a) HTN **(b)** SIN

Figure 5.15 Iteration and runtime for example 5 with respect to subdomain overlap.

Next, we test the effect of increasing the domain size on our nonlinear solver for example 5. We first test the serial algorithm for two sets of resolutions $h = 0.05, 0.01$ and four sets of limits $L = 0.5, 1.0, 1.5, 2.0$. The solution is computed up to a relative residual tolerance of 10^{-8}. The Krylov tolerance is set to 10^{-4} instead of the usual 10^{-5} due to convergence issues for the linear solver in the largest test case. The NK performance results are shown in Table 5.13. The error does increase with respect to increasing domain size, but not to the severity shown in example 1 Table 5.8. The computational runtime grows rapidly for $h = 0.01$ with respect to L. Domain limit $L = 1.5$ corresponds to 300×300 mesh, and $L = 2.0$ to 400×400 for $h = 0.01$.

Table 5.13 Results for the Serial Algorithm on Example 5

L	Iters	Error	Runtime
0.5	10	2.6200E-03	0.0141
1.0	12	4.9640E-03	0.0588
1.5	22	7.2660E-03	0.3500
2.0	23	9.9330E-03	0.6974

(a) $h = 0.05$

L	Iters	Error	Runtime
0.5	12	3.0370E-04	0.8329
1.0	24	5.4900E-04	17.1841
1.5	33	7.9540E-04	79.3232
2.0	44	1.0780E-03	237.8980

(b) $h = 0.01$

We next compare four-subdomain DDMs for $h = 0.01$ with $L = 1.5$ and $L = 2.0$. SIN is omitted because it did not converge within a reasonable time. We tune NKS and HTN through residual tolerances and overlap. Table 5.14 shows that both are faster than the serial method, with NKS giving lower runtimes and errors in these cases. The apparent superlinear speedup should be interpreted cautiously because the serial and parallel configurations use different Krylov tolerances; it does not isolate the effect of parallelism.

Table 5.14 DDM Performance on the Large Scale Problem

	NKS			HTN		
	Iterations	Error	Time (s)	Iterations	Error	Time (s)
L=1.5	33	8.085E-04	20.17	11	1.142E-03	36.15
L=2.0	36	1.166E-03	38.40	15	5.354E-03	97.94

We then test the DDM for the cases $L = 1.5$ and $L = 2.0$ for more subdomains. The SNK is run with $N_d = 4, 6, 8, 9$. The two resolutions $h = 0.05$ and $h = 0.01$ were run with a separate set of tuning parameters. The results for $h = 0.05$ are shown in Table 5.15, and for $h = 0.01$ in Table 5.16. On the finer resolution, the Newton tolerance needs to be stricter because the DDM residual would otherwise stagnate. Both cases were run with a global relative residual tolerance of 10^{-9}. The results indicate that both the iteration number and error are stable with respect to change in L from 1.5 to 2.0.

Table 5.15 Resolution $h = 0.05$, Overlap $p = 40\%$, Newton Tolerance 10^{-2}, and Krylov Tolerance 10^{-1}

	$N_d = 4$		$N_d = 6$		$N_d = 8$		$N_d = 9$	
L	Iter.	Error	Iter.	Error	Iter.	Error	Iter.	Error
1.5	15	7.263e-03	21	7.263e-03	30	7.263e-03	28	7.264e-03
2.0	17	9.921e-03	22	9.927e-03	31	9.924e-03	28	9.926e-03

Table 5.16 Resolution $h = 0.05$, Overlap $p = 10\%$, Newton Tolerance 10^{-3}, and Krylov Tolerance 10^{-1}

	$N_d = 4$		$N_d = 6$		$N_d = 8$		$N_d = 9$	
L	Iter.	Error	Iter.	Error	Iter.	Error	Iter.	Error
1.5	41	6.195e-04	61	5.752e-04	83	5.801e-04	80	5.661e-04
2.0	41	4.238e-04	60	2.424e-04	87	2.174e-04	79	2.298e-04

5.2.6 Comments on the Results

The PETSc implementation of the wide stencil scheme allowed us to compare parallel solvers for the 2D Dirichlet Monge-Ampère problem. The main methods, NKS and SNK, were benchmarked against serial Newton-Krylov using solutions with different regularity.

The experiments examined local solve accuracy, subdomain overlap, domain size, and scaling. Relaxing local Newton tolerances substantially improves SNK runtime. A relative Newton tolerance of 10^{-1} gives a useful balance in many tests. More aggressive under-solving can be faster, but may compromise convergence, particularly for larger problems.

Increasing the domain size at fixed spatial resolution increases the observed error, even for the serial solver. One possible contributor is the relative stopping criterion: the global nonlinear residual is reduced by a factor of 10^8 from its initial value. If that initial value grows with the domain, the final absolute residual can also grow. This may contribute to the larger solution error.

NKS favors zero overlap in the runtime tests, while the best SNK overlap ranges from 15% to 40%, depending on the problem. Both methods reduce runtime relative to the serial algorithm. NKS generally gives better strong scaling, except for example 3, whose boundary gradient singularity favors SNK. Neither method gives uniformly good weak scaling, although SNK with $N_d = 2$ exceeds 90% efficiency for some examples.

These experiments illustrate the numerical difficulty of the Dirichlet Monge-Ampère problem and the value of parallel solvers. NKS performs better on three of the four principal examples with relatively little tuning. SNK performs better on the example with a boundary gradient singularity, which is also the most difficult case for the serial solver. The scaling results remain problem dependent.

5.3 Concluding Remarks

The original project began as a graduate course investigation of domain decomposition for optimal transport. Its central question was whether nonlinear Schwarz would converge for the Monge-Ampère equation. The results support convergence of overlapping Schwarz for the Dirichlet problem, with monotone wide stencil discretization playing a key role.

In addition to standard DDM parameters, we varied solution regularity. The examples include a smooth solution, solutions with vanishing Hessian eigenvalues, and a solution with a boundary gradient singularity. The last is the most difficult to solve numerically, possibly because of its effect on Newton's Jacobian systems.

Two findings suggest further research. First, nonlinear Schwarz requires careful control of local work. A speedup was obtained only after limiting local solve accuracy. Local solvers may otherwise spend too much time solving problems with inaccurate neighboring data, while frequent exchanges compensate partly for the lack of global guidance. Using Schwarz as a nonlinear preconditioner, or combining it with nonlinear multigrid or nonlinear GMRES, may provide better globalization.

Second, nonlinear Schwarz outperforms NKS on the boundary-singularity example, although NKS is faster and easier to tune on the other three principal examples. Both methods use Newton iteration, globally in NKS and locally in SNK. This suggests that isolating a difficult region may help the nonlinear solve. A useful

next step is to quantify how a singularity's location relative to the decomposition affects Schwarz convergence.

APPENDIX A

HELMHOLTZ

This appendix collects supplementary calculations for the Helmholtz equation.

A.1 Fundamental Exterior Problem

Consider the exterior Helmholtz problem with Robin boundary conditions on the scattering surface. When the scatterer is a cylinder and the incident wave is a plane wave, the solution can be written explicitly. Figure A.1 shows the geometry. We write the reflection condition as $\beta_1 u^{\text{total}} + \partial_{\mathbf{n}} u^{\text{total}} = 0$, where $u^{\text{total}} = u + u^{\text{inc}}$ represents the amplitude of the total wave field. u is the unknown scattered wave and u^{inc} is the given incident wave. We will assume β_1 is a complex scalar. The BVP is expressed as follows:

$$\begin{cases} \Delta u + k^2 u = 0 & x \in \Omega \\ \beta_1 u + \partial_{\mathbf{n}} u = -\beta_1 u^{\text{inc}} - \partial_{\mathbf{n}} u^{\text{inc}} & x \in \Gamma \\ |x|^{1/2} \left(\nabla u \cdot \dfrac{x}{|x|} - \mathrm{i}ku \right) = 0 & |x| \to +\infty. \end{cases} \tag{A.1}$$

For a circular boundary Γ of radius r_1 centered at the origin, the solution can be expanded in angular Fourier modes. Using polar coordinates $x_1 = r\cos\theta, x_2 = r\sin\theta$, write

$$u(r,\theta) = \sum_{m=-\infty}^{\infty} C_m H_m^{(1)}(kr) e^{\mathrm{i}m\theta}, \qquad r_1 \le r, 0 \le \theta < 2\pi \tag{A.2}$$

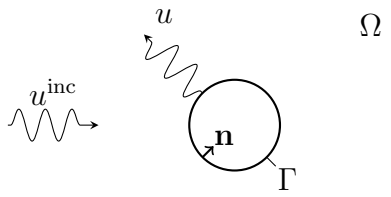


Figure A.1 Exterior Scattering problem.

where $H_m^{(1)}$ is the Hankel function of the 1st kind of order m, and A_m are coefficients to be determined. Note that this general form satisfies the Helmholtz equation and the Sommerfeld condition.

A.1.1 Plane Wave

We assume that the incident wave u^{inc} is a plane wave. Let the components of the wavevector $\mathbf{k}$ be written as $k_1 = k\cos(\theta_{\text{inc}})$ and $k_2 = k\sin(\theta_{\text{inc}})$, where the angle of incidence θ_{inc} is a known quantity. In polar coordinates, we may write the plane wave as

$$\begin{aligned} u^{\text{inc}} &= e^{-\mathrm{i}\mathbf{k}\cdot x} \\ &= e^{-\mathrm{i}kr(\cos\theta\cos\theta_{\text{inc}}+\sin\theta\sin\theta_{\text{inc}})} \\ &= e^{-\mathrm{i}kr(\cos(\theta-\theta_{\text{inc}}))}. \end{aligned} \tag{A.3}$$

Introduce the shifted angle $\phi = \theta - \theta_{\text{inc}}$. To account for $\theta_{\text{inc}} \neq 0$ in the ansatz Equation (A.2), write

$$u = \sum_{m=-\infty}^{\infty} \tilde{C}_m H_m^{(1)}(kr) e^{im\phi}. \tag{A.4}$$

Similarly, expand the incident wave in a Fourier series in ϕ:

$$u^{\text{inc}} = \sum_{m=-\infty}^{\infty} D_m e^{\mathrm{i}m\phi} \tag{A.5}$$

where the standard Fourier coefficients D_m, depending on kr ,are given by the integral

$$D_m(kr) = \frac{1}{2\pi}\int_0^{2\pi} e^{-\mathrm{i}kr\cos\phi} e^{-im\phi}\mathrm{d}\phi \tag{A.6}$$

Jackson [47] gives this particular integral representation of the Bessel function of the first kind

$$J_m(x) = \frac{1}{2\pi i^m}\int_0^{2\pi} e^{\mathrm{i}x\cos\phi - \mathrm{i}m\phi}\mathrm{d}\phi.$$

Using this formula, the Fourier coefficient can be expressed in terms of the Bessel functions:

$$\begin{aligned} D_m &= \frac{\mathrm{i}^m}{2\pi \mathrm{i}^m} \int_0^{2\pi} e^{\mathrm{i}(-kr)\cos\phi - \mathrm{i}m\phi} \mathrm{d}\phi \\ &= (\mathrm{i})^m J_m(-kr) \\ &= (-\mathrm{i})^m J_m(kr) \end{aligned} \tag{A.7}$$

Hence, the angular Fourier expansion of the plane wave is

$$u^{\mathrm{inc}} = \sum_{m=-\infty}^{\infty} (-\mathrm{i})^m J_m(kr) e^{\mathrm{i}m\phi} \tag{A.8}$$

Bessel functions satisfy a connection formula $J_{-m}(z) = (-1)^m J_m(z)$. Therefore, D_m is symmetric with respect to m, i.e. $D_{-m} = D_m$. Hence, we can express the incident wave alternatively as

$$\begin{aligned} u^{\mathrm{inc}} &= J_0(kr) + \sum_{m=1}^{\infty} (-\mathrm{i})^m J_m(kr) \left(e^{\mathrm{i}m\phi} + e^{-\mathrm{i}m\phi} \right) \\ &= J_0(kr) + \sum_{m=1}^{\infty} (-\mathrm{i})^m J_m(kr) 2\cos(m\phi) \\ &= \sum_{m=0}^{\infty} \epsilon_m (-\mathrm{i})^m J_m(kr)\cos(m\phi), \end{aligned} \tag{A.9}$$

where $\epsilon_m = 1$ if $m = 0$ and $\epsilon_m = 2$ for $m \geq 1$.We may take the radial derivative of Equation (A.9) to get

$$\frac{\partial}{\partial r} u^{\mathrm{inc}} = \sum_{m=0}^{\infty} \epsilon_m (-\mathrm{i})^m k \dot{J}_m(kr)\cos(m\phi). \tag{A.10}$$

For clarification, the symbol $\dot{J}_m(kr)$ means $\frac{d}{dx} J_m(x)|_{x=kr_1}$. We may use the formula

$$\dot{J}_m(kr) = \frac{1}{2} \left(J_{m-1}(kr) - J_{m+1}(kr) \right) \tag{A.11}$$

to compute the derivative. We may use the expression above to easily prove the connection formula $\dot{J}_{-m}(kr) = (-1)^m \dot{J}_m(kr)$.

A.1.2 Exterior Solution

The general solution and incident wave are now expressed using Bessel functions. Substituting (A.4) and (A.8) into the Robin condition in (A.1) gives

$$\tilde{C}_m = -(-\mathrm{i})^m \frac{-\beta_1 J_m(kr_1) + k\dot{J}_m(kr_1)}{-\beta_1 H_m^{(1)}(kr_1) + k\dot{H}_m^{(1)}(kr_1)}. \tag{A.12}$$

For a circular scatterer Γ of radius r_1 centered at the origin and incident wave $u^{\mathrm{inc}} = e^{-\mathrm{i}kr(\cos(\theta-\theta_{\mathrm{inc}}))}$, the exterior Robin solution is

$$u(r,\theta) = \sum_{m=-\infty}^{\infty} -(-\mathrm{i})^m \frac{-\beta_1 J_m(kr_1) + k\dot{J}_m(kr_1)}{-\beta_1 H_m^{(1)}(kr_1) + k\dot{H}_m^{(1)}(kr_1)} H_m^{(1)}(kr) e^{\mathrm{i}m(\theta-\theta_{\mathrm{inc}})} \tag{A.13}$$

for $r_1 \le r, 0 \le \theta < 2\pi$.

The coefficients satisfy $\tilde{C}_m = (-1)^m \tilde{C}_{-m}$, while $H_m^{(1)} = (-1)^m H_{-m}^{(1)}$. These relations allow the series to be evaluated using only nonnegative indices. Equation (A.13) becomes

$$u(r,\theta) = \sum_{m=0}^{\infty} -\epsilon_m (-\mathrm{i})^m \frac{-\beta_1 J_m(kr_1) + k\dot{J}_m(kr_1)}{-\beta_1 H_m^{(1)}(kr_1) + k\dot{H}_m^{(1)}(kr_1)} H_m^{(1)}(kr)\cos(m(\theta-\theta_{\mathrm{inc}})). \tag{A.14}$$

A.2 Scattering Problem on the Annulus

We next consider the annulus problem. It differs from (A.1) by truncating the exterior domain at a circle Σ enclosing the scatterer Γ. The computational domain is $\Omega = \{r_1 < |x| < r_2\}$, bounded by Γ and Σ; see Figure A.2. The BVP is given in (A.15).

$$\begin{cases} \Delta u + k^2 u = 0 & r_1 < |x| < r_2 \\ \beta_1 u + \partial_{\mathbf{n}} u = -\beta_1 u^{\mathrm{inc}} - \partial_{\mathbf{n}} u^{\mathrm{inc}} & |x| = r_1 \\ \beta_2 u + \partial_{\mathbf{n}} u = 0 & |x| = r_2 \end{cases} \tag{A.15}$$

For $\beta_2 = -\mathrm{i}k$, this problem is equivalent to solving the exterior Helmholtz equation with a 1st order absorbing boundary condition (ABC). The Sommerfeld radiation condition in (A.1) has been approximated by a condition on an artificial circle Σ

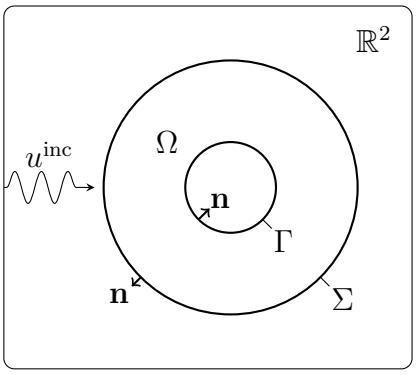


Figure A.2 Exterior scattering problem truncated by an artificial surface Σ.

which is a finite distance away from the scatterer. In this sense, we can think of the annulus problem as an approximation to the exterior problem controlled by the distance $r_2 - r_1$. As r_2 approaches infinity, the solution to (A.15) converges to the solution of the exterior problem expressed in (A.1).

We will solve the annulus problem by presenting an ansatz:

$$u(r,\theta) = \sum_{m=-\infty}^{\infty} (A_m J_m(kr) + B_m Y_m(kr)) e^{\mathrm{i}m\theta}. \tag{A.16}$$

Here J_m and Y_m are Bessel functions of the first and second kinds of order m, and A_m, B_m are coefficients to be determined. This ansatz satisfies the Helmholtz equation and follows from separation of variables, as outlined below.

In polar coordinates, the Helmholtz equation is

$$u_{rr} + \frac{1}{r}u_r + \frac{1}{r^2}u_{\theta\theta} + k^2 u = 0. \tag{A.17}$$

Next, we let $u = R(r)\Theta(\theta)$. If we introduce a matching constant m^2, we get the set of equations

$$\begin{cases} \Theta'' + m^2\Theta = 0 \\ r^2 R'' + rR' + (r^2k^2 - m^2)R = 0. \end{cases} \tag{A.18}$$

Periodicity gives the angular modes $\Theta_m(\theta) = e^{im\theta}$. The radial equation has solutions $R_m(r) = c_1 J_m(kr) + c_2 Y_m(kr)$. Superposition, $u(r,\theta) = \sum_m R_m(r)\Theta_m(\theta)$, yields the ansatz (A.16).

Next, we must use the conditions on Γ and Σ as well as the plane wave expansion in (A.8) to determine the coefficients. We choose the incident wave to be the plane wave $u^{\text{inc}} = e^{-\mathrm{i}\mathbf{k}\cdot x}$. The condition on Γ gives us

$$\beta_1\Big(A_m J_m(kr_1) + B_m Y_m(kr_1)\Big) + A_m k\dot{J}_m(kr_1) + B_m k\dot{Y}_m(kr_1) = \\ -(-\mathrm{i})^m e^{-im\theta_{\text{inc}}}\Big(\beta_1 J_m(kr_1) + k\dot{J}_m(kr_1)\Big). \quad \text{(A.19)}$$

The condition on Σ gives us

$$\beta_2\Big(A_m J_m(kr_2) + B_m Y_m(kr_2)\Big) + A_m k\dot{J}_m(kr_2) + B_m k\dot{Y}_m(kr_2) = 0. \quad \text{(A.20)}$$

A linear system for the coefficients can be written compactly in the form:

$$\begin{bmatrix} 1 & \gamma_m^{(1)} \\ 1 & \gamma_m^{(2)} \end{bmatrix}\begin{bmatrix} A_m \\ B_m \end{bmatrix} = \begin{bmatrix} -(-i)^m e^{-\mathrm{i}m\theta_{\text{inc}}} \\ 0 \end{bmatrix} \quad \text{(A.21)}$$

where

$$\gamma_m^{(i)} = \frac{\beta_i Y_m(kr_i) + k\dot{Y}_m(kr_i)}{\beta_i J_m(kr_i) + k\dot{J}_m(kr_i)}, \qquad i = 1, 2. \quad \text{(A.22)}$$

We have an explicit solution for the weights using Cramer's Rule:

$$A_m = e^{-im\theta_{\text{inc}}}\frac{(-i)^m\gamma_m^{(2)}}{\gamma_m^{(1)} - \gamma_m^{(2)}}, \quad B_m = e^{-im\theta_{\text{inc}}}\frac{-(-i)^m}{\gamma_m^{(1)} - \gamma_m^{(2)}}. \quad \text{(A.23)}$$

Finally, we arrive at the explicit solution for the annulus problem

$$u(r,\theta) = \sum_{m=-\infty}^{\infty}\left(\frac{(-\mathrm{i})^m\gamma_m^{(2)}}{\gamma_m^{(1)} - \gamma_m^{(2)}}J_m(kr) + \frac{-(-\mathrm{i})^m}{\gamma_m^{(1)} - \gamma_m^{(2)}}Y_m(kr)\right)e^{\mathrm{i}m(\theta-\theta_{\text{inc}})}. \quad \text{(A.24)}$$

A.3 Generalized Problem and the Bilinear Form

Let us now focus on a generalization of the annulus problem (A.15). We do not require Γ and Σ to be circles. They are non-overlapping sufficiently-smooth, closed contours such that Γ is contained in the interior of Σ. The computational domain is the bounded, 2D area in between the curves Σ and Γ. The BVP is expressed in $(A.25)$.

$$\begin{cases} \Delta u + k^2 u = 0 & x \in \Omega \\ \beta_1 u + \partial_{\mathbf{n}} u = -\beta_1 u^{\text{inc}} - \partial_{\mathbf{n}} u^{\text{inc}} & x \in \Gamma \\ \beta_2 u + \partial_{\mathbf{n}} u = 0 & x \in \Sigma \end{cases} \tag{A.25}$$

An explicit solution is generally unavailable. Since Ω is bounded, FEM can approximate u. We first express (A.25) in bilinear form.

Multiply each equation in (A.25) by a test function $v \in H^1$ and integrate over the corresponding domain:

$$\begin{aligned} &\int_\Omega v\Delta u \,\mathrm{d}x + \int_\Omega k^2 uv \,\mathrm{d}x = 0 \\ &\int_\Gamma \partial_{\mathbf{n}} u \, v \,\mathrm{d}s = -\int_\Gamma \beta_1 u \, v \,\mathrm{d}s - \int_\Gamma (\beta_1 u^{\text{inc}} \, v + \partial_{\mathbf{n}} u^{\text{inc}} \, v) \,\mathrm{d}s \\ &\int_\Sigma \partial_{\mathbf{n}} u \, v \,\mathrm{d}s = -\int_\Sigma \beta_2 u \, v \,\mathrm{d}s \end{aligned} \tag{A.26}$$

For the first integral in (A.26), we can use divergence product rule, $\nabla \cdot (v\nabla u) = \nabla u \cdot \nabla v + v\nabla \cdot \nabla u$, to expand it. Then we can use divergence rule to change the volume integral into a surface integral. Note that the boundary of Ω is the union $\partial\Omega = \Sigma \cup \Gamma$. Because we chose to define $\mathbf{n}$ as the unit normal always pointing *out* of Ω, Σ is oriented counter clockwise and Γ is oriented clockwise. Thus we rewrite the

area integral like so:

$$\begin{aligned}\int_\Omega v\Delta u\,\mathrm{d}x &= -\int_\Omega \nabla u\cdot\nabla v\,\mathrm{d}x + \int_\Omega \nabla\cdot(v\nabla u)\,\mathrm{d}x\\ &= -\int_\Omega \nabla u\cdot\nabla v\,\mathrm{d}x + \int_{\partial\Omega}(v\nabla u)\cdot\mathbf{n}\,\mathrm{d}s\\ &= -\int_\Omega \nabla u\cdot\nabla v\,\mathrm{d}x + \int_\Sigma v(\nabla u\cdot\mathbf{n})\,\mathrm{d}s + \int_\Gamma v(\nabla u\cdot\mathbf{n})\,\mathrm{d}s\\ &= -\int_\Omega \nabla u\cdot\nabla v\,\mathrm{d}x + \int_\Sigma \partial_{\mathbf{n}}u\,v\,\mathrm{d}s + \int_\Gamma \partial_{\mathbf{n}}u\,v\,\mathrm{d}s\end{aligned} \tag{A.27}$$

Combining (A.26) and (A.27) gives the bilinear formulation

$$\int_\Omega \nabla u\cdot\nabla v\,\mathrm{d}x - \int_\Omega k^2uv\,\mathrm{d}x + \int_\Gamma \beta_1 u\,v\,\mathrm{d}s + \int_\Sigma \beta_2 u\,v\,\mathrm{d}s = -\int_\Gamma(\beta_1 u^{\mathrm{inc}} + \partial_{\mathbf{n}}u^{\mathrm{inc}})v\,\mathrm{d}s. \tag{A.28}$$

This equation holds for general linear operators β_1, β_2. Let us define the left-hand side as the bilinear operator

$$a(u,v) := \int_\Omega \nabla u\cdot\nabla v\,\mathrm{d}x - \int_\Omega k^2uv\,\mathrm{d}x + \int_\Gamma \beta_1 u\,v\,\mathrm{d}s + \int_\Sigma \beta_2 u\,v\,\mathrm{d}s. \tag{A.29}$$

If the boundary operators are scalar coefficients, the bilinear form is symmetric, i.e., $a(u,v) = a(v,u)$.

A.4 Laplace-Beltrami on the Interface

In (A.28), suppose we choose the Σ operator to have the form $\beta_2 = -\mathrm{i}k + \gamma\Delta_\Sigma$, where γ is a scalar parameter. β_1 can be any scalar parameter. The Laplace-Beltrami operator Δ_S is a generalization of the usual Euclidean Laplace operator $\Delta = \sum_{i=1,d}\partial_{ii}$, where d is the dimension. The second derivatives for Δ_Σ are instead taken with respect to each orthogonal direction the manifold Σ. In our context, the second derivatives are taken in the normal and tangential direction along a curve in 2D, hence $\Delta_S = \partial_{\mathbf{n}}^2 + \partial_\tau^2$.

The Laplace–Beltrami contribution can be rewritten using integration by parts on Σ. The endpoint contribution vanishes because Σ is a closed contour, restoring a

symmetric representation of the bilinear form.

$$\begin{aligned}\int_\Sigma \beta_2 u\, v \,\mathrm{d}s &= \int_\Sigma (-\mathrm{i}k + \gamma\Delta_\Sigma) u\, v \,\mathrm{d}s \\ &= -\mathrm{i}k \int_\Sigma uv \,\mathrm{d}s + \int_\Sigma \gamma \left(-\nabla_\Sigma u \cdot \nabla_\Sigma v + \nabla_\Sigma \cdot (v \nabla_\Sigma u)\right) \mathrm{d}s \\ &= -\mathrm{i}k \int_\Sigma uv \,\mathrm{d}s - \gamma \int_\Sigma \nabla_\Sigma u \cdot \nabla_\Sigma v \,\mathrm{d}s + \gamma \int_{\partial\Sigma} (v \nabla_\Sigma u) \cdot \mathbf{n} \,\mathrm{d}s \\ &= -\mathrm{i}k \int_\Sigma uv \,\mathrm{d}s - \gamma \int_\Sigma \nabla_\Sigma u \cdot \nabla_\Sigma v \,\mathrm{d}s\end{aligned} \tag{A.30}$$

To see why the endpoint term vanishes, parameterize Σ by $s(t)$, $a \le t \le b$. Because the curve is closed, $s(a) = s(b)$, and the two endpoint contributions cancel.

$$\int_{\partial\Sigma} (v \nabla_\Sigma u) \cdot \mathbf{n} \,\mathrm{d}s = v(s(t)) \nabla_\Sigma u(s(t)) \cdot \mathrm{n}(s(t)) \Big|_{t=a}^{t=b} = 0 \tag{A.31}$$

If Σ has distinct endpoints, the endpoint contribution in (A.31) need not vanish. This issue arises when domain decomposition introduces open interfaces.

A.4.1 Laplace-Beltrami with Domain Decomposition

Consider the annulus problem where the domain is divided into three equal subdomains, $\Omega_1, \Omega_2, \Omega_3$. See figure A.3 for the configuration. For simplicity we will let the scattering coefficient $\beta_1 = 0$. We wish to apply the NDDM algorithm to solve the Helmholtz equation and use the Laplace-Beltrami transmission operator on the artificial interfaces Σ_{ij}. There is some subtlety involved in defining the auxiliary unknowns.

We first specify the interface orientation. Define $\bar{\Sigma}_{ij} = \bar{\Omega}_i \cap \bar{\Omega}_j$, so $\Sigma_{ij} = \Sigma_{ji}$ as sets. Each subdomain boundary $\partial\Omega_i$ is positively oriented. Define the surface gradient by

$$\nabla_{\partial\Omega_i} := (\mathbf{n}_i, \tau_i), \tag{A.32}$$

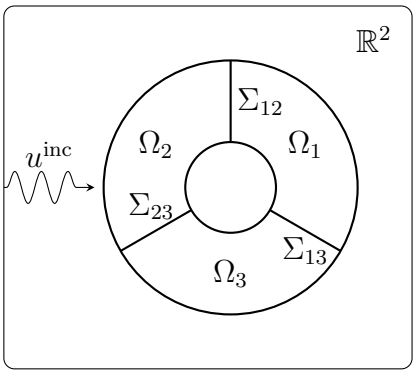


Figure A.3 A domain with three subdomains.

which is simply the tuple consisting of the normal and tangential directional derivatives with respect to the local domain. Then we may define $\nabla_{\Sigma_{ij}}$ to mean $\nabla_{\partial\Omega_i}$ restricted to Σ_{ij}. Although, we have the property that $\nabla_{\Sigma_{ij}} u_i = -\nabla_{\Sigma_{ji}} u_j$ on the interface Σ_{ij}, the Laplace-Beltrami does not have this negative property. This is because the Laplace-Beltrami is defined as the inner product of the surface gradients so the double negative cancels out. Therefore we have the symmetry property $\Delta_{\Sigma_{ij}} u_i = \Delta_{\Sigma_{ji}} u_j$ along the boundary Σ_{ij}.

Now we can discuss the condition along the interface. To ensure that the local solutions u_i and u_j are at least C^1 along the common interface, we enforce

$$\begin{cases} -\mathrm{i}ku_i + \gamma\Delta_{\Sigma_{ij}} u_i = -\mathrm{i}ku_j + \gamma\Delta_{\Sigma_{ji}} u_j \\ \partial_{\mathbf{n}_i} u_i = -\partial_{\mathbf{n}_j} u_j \end{cases}, \qquad \text{on } \Sigma_{ij}. \tag{A.33}$$

Now, we will rewrite the continuity condition as the transmission conditions:

$$\begin{cases} -\mathrm{i}ku_i + \gamma\Delta_{\Sigma_{ij}} u_i + \partial_{\mathbf{n}_i} u_i = -\mathrm{i}ku_j + \gamma\Delta_{\Sigma_{ji}} u_j - \partial_{\mathbf{n}_j} u_j \\ -\mathrm{i}ku_i + \gamma\Delta_{\Sigma_{ij}} u_i - \partial_{\mathbf{n}_i} u_i = -\mathrm{i}ku_j + \gamma\Delta_{\Sigma_{ji}} u_j + \partial_{\mathbf{n}_j} u_j \end{cases}, \qquad \text{on } \Sigma_{ij}, \tag{A.34}$$

which is equivalent to (A.33). Next, we define our auxiliary unknowns as

$$\begin{aligned} g_{ij}(u_j) &= -\mathrm{i}ku_j + \gamma\Delta_{\Sigma_{ji}} u_j - \partial_{\mathbf{n}_j} u_j \\ g_{ji}(u_i) &= -\mathrm{i}ku_i + \gamma\Delta_{\Sigma_{ij}} u_i - \partial_{\mathbf{n}_i} u_i. \end{aligned} \tag{A.35}$$

Note that g_{ij} shall be interpreted as the boundary data which is supplied to domain Ω_i and is dependent on the solution from domain Ω_j. The local BVP on domain Ω_i can now be expressed as

$$\begin{cases} \Delta u_i + k^2 u_i = 0 & x \in \Omega_i \\ \beta u_i + \partial_{\mathbf{n}_i} u_i = -\beta u^{\text{inc}} - \partial_{\mathbf{n}} u^{\text{inc}} & x \in \Gamma_i \\ -\mathrm{i}k u_i + \gamma \Delta_{\Sigma_{ij}} u_i + \partial_{\mathbf{n}_i} u_i = g_{ij} & x \in \Sigma_{ij} : j \in \Lambda_i. \end{cases} \tag{A.36}$$

It should be understood that $g_{i0} = 0$ and the set of integers Λ_i is augmented to include 0 in addition to the neighbors of Ω_i. The boundary of each subdomain Ω_i consists of the sum of 4 interfaces: Γ_i, Σ_{i0}, and 2 more for each of its neighbors. For example, $\partial\Omega_1 = \Gamma_1 + \Sigma_{10} + \Sigma_{12} + \Sigma_{13}$. Take a test function $v_i \in H^1$, multiply it to each equation in (A.36), and then integrate each equation over their respective domains. We obtain the following equations:

$$\begin{aligned} &\int_{\Omega_i} (\nabla u_i \cdot \nabla v_i - k^2 u_i v_i) \mathrm{d}x = \int_{\partial\Omega_i} \partial_{\mathbf{n}_i} u_i \, v_i \, \mathrm{d}s \\ &\int_{\Gamma_i} \partial_{\mathbf{n}_i} u_i \, v_i \, \mathrm{d}s = -\int_{\Gamma_i} \beta u_i \, v_i \, \mathrm{d}s - \int_{\Gamma_i} (\beta u^{\text{inc}} + \partial_{\mathbf{n}_i} u^{\text{inc}}) v_i \mathrm{d}s \\ &\int_{\Sigma_{ij}} \partial_{\mathbf{n}_i} u_i \, v_i \, \mathrm{d}s = \int_{\Sigma_{ij}} g_{ij} v_i \, \mathrm{d}s - \int_{\Sigma_{ij}} (-\mathrm{i}k + \gamma \Delta_{\Sigma_{ij}}) u_i \, v_i \, \mathrm{d}s, \qquad \forall j \in \Lambda_i \end{aligned} \tag{A.37}$$

Consider the Laplace–Beltrami integral. Integration by parts now produces endpoint terms because the interface is open. Parameterize Σ_{ij} by $s(t)$, $a \le t < b$, in the counterclockwise orientation, and write $s_b = s(b)$, $s_a = s(a)$. Then

$$\begin{aligned} \int_{\Sigma_{ij}} (-\mathrm{i}k + \gamma \Delta_{\Sigma_{ij}}) u_i \, v_i \, \mathrm{d}s &= -\mathrm{i}k \int_{\Sigma_{ij}} u_i v_i \, \mathrm{d}s + \gamma \int_{\Sigma_{ij}} \Delta_{\Sigma_{ij}} u_i \, v_i \, \mathrm{d}s \\ &= -\mathrm{i}k \int_{\Sigma_{ij}} u_i v_i \, \mathrm{d}s - \gamma \int_{\Sigma_{ij}} \nabla_{\Sigma_{ij}} u_i \cdot \nabla_{\Sigma_{ij}} v_i \, \mathrm{d}s + \gamma \int_{\partial\Sigma_{ij}} (v \nabla_\Sigma u) \cdot \mathbf{n} \, \mathrm{d}s \\ &= -\int_{\Sigma_{ij}} \left(\mathrm{i}k u_i v_i + \gamma \nabla_{\Sigma_{ij}} u_i \cdot \nabla_{\Sigma_{ij}} v_i \right) \mathrm{d}s + \gamma (v_i(s) \partial_{\mathbf{n}} u_i(s)) \Big|_{s_a}^{s_b}. \end{aligned}$$

(A.38)

Choose H_0^1 here to denote the test subspace of H^1 whose functions vanish at cross-points. The endpoint contributions then vanish. Substituting the contour integrals into the PDE integral gives

$$\begin{aligned}\int_{\Omega_i} (\nabla u_i \cdot \nabla v_i - k^2 u_i v_i)\mathrm{d}x &= \int_{\partial\Omega_i} \partial_{\mathbf{n}_i} u_i \, v_i \, \mathrm{d}s \\ &= \int_{\Gamma_i} \partial_{\mathbf{n}_i} u_i \, v_i \, \mathrm{d}s + \sum_{j\in\Lambda_i} \int_{\Sigma_{ij}} \partial_{\mathbf{n}_i} u_i \, v_i \, \mathrm{d}s \\ &= -\int_{\Gamma_i} \beta u_i \, v_i \, \mathrm{d}s - \int_{\Gamma_i} (\beta u^{\mathrm{inc}} + \partial_{\mathbf{n}_i} u^{\mathrm{inc}}) v_i \mathrm{d}s + \\ &\qquad \sum_{j\in\Lambda_i} \left(\int_{\Sigma_{ij}} g_{ij} v_i \, \mathrm{d}s + \int_{\Sigma_{ij}} \left(\mathrm{i}k u_i v_i + \gamma \nabla_{\Sigma_{ij}} u_i \cdot \nabla_{\Sigma_{ij}} v_i \right) \mathrm{d}s \right).\end{aligned} \tag{A.39}$$

Finally, we arrive at our bilinear form

$$a_i(u_i, v_i) = \ell_i(v_i; g_{ij}), \tag{A.40}$$

where

$$\begin{aligned}a_i(u_i, v_i) = \int_{\Omega_i} (\nabla u_i \cdot \nabla v_i - k^2 u_i v_i)\mathrm{d}x + \int_{\Gamma_i} \beta u_i \, v_i \, \mathrm{d}s - \\ \sum_{j\in\Lambda_i} \int_{\Sigma_{ij}} (\mathrm{i}k u_i v_i + \gamma \nabla_{\Sigma_{ij}} u \cdot \nabla_{\Sigma_{ij}} v)\mathrm{d}s\end{aligned} \tag{A.41}$$

and

$$\ell_i(v_i; g_{ij}) = -\int_{\Gamma_i} \left(\beta u^{\mathrm{inc}} + \partial_{\mathbf{n}_i} u^{\mathrm{inc}}\right) v_i \, \mathrm{d}s + \sum_{j\in\Lambda_i} \int_{\Sigma_{ij}} g_{ij} v_i \, \mathrm{d}s. \tag{A.42}$$

The weak problem seeks $u_i \in H^1$ satisfying (A.40) for every $v_i \in H_0^1$.

A.5 Weak NDDM-FEM Algorithm

We wish to formally write the continuous version of the FEM-NDDM algorithm to solve the original annulus problem (A.15). Let n be the index of iteration. At each

iteration, we need to obtain the weak solution $u_i^{(n)}$ depending on the data $g_{ij}^{(n-1)}$, for each domain Ω_i. This is to say, we equip the boundary conditions

$$\begin{aligned} -\mathrm{i}ku_i^{(n)} + \gamma\Delta_{\Sigma_{ij}}u_i^{(n)} + \partial_{\mathbf{n}_i}u_i^{(n)} &= g_{ij}^{(n-1)} \qquad \text{for domain } \Omega_i, \\ -\mathrm{i}ku_j^{(n)} + \gamma\Delta_{\Sigma_{ji}}u_j^{(n)} + \partial_{\mathbf{n}_j}u_j^{(n)} &= g_{ji}^{(n-1)} \qquad \text{for domain } \Omega_j. \end{aligned} \tag{A.43}$$

After solving for the solution $u_i^{(n)}$, we may use it to update the next set of boundary data. The next set of boundary data is defined to be the auxiliary functions (A.35) evaluated with the current solution as an input. With some algebraic manipulation, we can use equations (A.35) and (A.43) to obtain the following recurrence relations:

$$\begin{aligned} g_{ji}^{(n)} := g_{ji}(u_i^{(n)}) &= -\mathrm{i}ku_i^{(n)} + \gamma\Delta_{\Sigma_{ij}}u_i^{(n)} - \partial_{\mathbf{n}_i}u_i^{(n)} \\ &= -\mathrm{i}ku_i^{(n)} + \gamma\Delta_{\Sigma_{ij}}u_i^{(n)} - \left(g_{ij}^{(n-1)} + \mathrm{i}ku_i^{(n)} - \gamma\Delta_{\Sigma_{ij}}u_i^{(n)}\right) \\ &= -2\mathrm{i}ku_i^{(n)} + 2\gamma\Delta_{\Sigma_{ij}}u_i^{(n)} - g_{ij}^{(n-1)} \\ g_{ij}^{(n)} := g_{ij}(u_j^{(n)}) &= -\mathrm{i}ku_j^{(n)} + \gamma\Delta_{\Sigma_{ji}}u_j^{(n)} - \partial_{\mathbf{n}_j}u_j^{(n)} \\ &= -\mathrm{i}ku_j^{(n)} - \left(g_{ji}^{(n-1)} + \mathrm{i}ku_j^{(n)} - \gamma\Delta_{\Sigma_{ji}}u_j^{(n)}\right) \\ &= -2\mathrm{i}ku_j^{(n)} + 2\gamma\Delta_{\Sigma_{ji}}u_j^{(n)} - g_{ji}^{(n-1)} \end{aligned} \tag{A.44}$$

Equation (A.44) is the update formula, and it involves the Laplace-Beltrami. We must express the update formula in bilinear form before we finally write the NDDM algorithm. We test (A.44) with function $v_j \in H_0^1$ and integrate:

$$\begin{aligned} \int_{\Sigma_{ij}} g_{ij}^{(n)} v_j \,\mathrm{d}s &= \int_{\Sigma_{ij}} \left(-2\mathrm{i}ku_j^{(n)} + 2\gamma\Delta_{\Sigma_{ji}}u_j^{(n)} - g_{ji}^{(n-1)}\right) v_j \,\mathrm{d}s \\ &= -2\mathrm{i}k\int_{\Sigma_{ij}} u_j^{(n)} v_j \,\mathrm{d}s - 2\gamma\int_{\Sigma_{ij}} \nabla_{\Sigma_{ij}}u_j^{(n)} \cdot \nabla_{\Sigma_{ij}} v_j \,\mathrm{d}s - \int_{\Sigma_{ij}} g_{ji}^{(n-1)} v_j \,\mathrm{d}s \end{aligned} \tag{A.45}$$

Algorithm 11: NDDM in the weak form.

Initialize all $g_{ij}^{(0)}$;
for *each domain* i **do**
  Compute the scattering RHS $-\int_{\Gamma_i}(\beta u^{\text{inc}} + \partial_{\mathbf{n}_i} u^{\text{inc}})\, v_i \,\mathrm{d}s$;
end
Initialize iteration index $n = 1$, residual $R = 1$;
Choose a tolerance $\epsilon < 1$;
while $R > \epsilon$ **do**
  for *each domain* i **do**
    Find $u_i^{(n)} \in H^1$ such that $\forall v_i \in H_0^1$, $a_i\left(u_i^{(n)}, v_i\right) = \ell_i\left(v_i; g_{ij}^{(n-1)}\right)$;
  end
  for *each domain* i **do**
    Calculate all updates $\int_{\Sigma_{ij}} g_{ji}^{(n)} v_i \,\mathrm{d}s =$
     $-2\mathrm{i}k\int_{\Sigma_{ij}} u_i^{(n)} v_i \,\mathrm{d}s - 2\gamma\int_{\Sigma_{ij}} \nabla_{\Sigma_{ij}} u_i^{(n)} \cdot \nabla_{\Sigma_{ij}} v_i \,\mathrm{d}s - \int_{\Sigma_{ij}} g_{ij}^{(n-1)} v_i \,\mathrm{d}s$;
  end
  Calculate $R = \sum_i \left\| g_{ij}^{(n)} - g_{ij}^{(n-1)} \right\|$;
  n++;
end

Discretizing this algorithm gives the FEM-NDDM formulation discussed in the main text. The next section derives the corresponding method for the square-root transmission operator.

A.6 Domain Decomposition with the Square Root Operator

We now derive FEM-NDDM with a square-root transmission operator on the geometry in Figure A.3. We begin with the local BVP, using the following operator on the interface S:

$$\beta^{N_p,\alpha,\varepsilon}\{u\} = \begin{cases} -\mathrm{i}k\left(C_0 u + \dfrac{1}{k_\varepsilon^2}\displaystyle\sum_{\ell=1}^{N_p} A_\ell \Delta_S \varphi_\ell\right), \\ \left(1 + \dfrac{1}{k_\varepsilon^2} B_\ell \Delta_S\right)\varphi_\ell = u, \qquad\qquad \forall \ell = 1, \cdots, N_p. \end{cases} \tag{A.46}$$

We assume the ε parameter is independent of S thus making $k_\varepsilon^2 := (k + \mathrm{i}\varepsilon)^2$ an ordinary complex scalar. The other parameters N_p and α are arbitrary. We express

the continuity for solutions u_i and u_j along the common interface as

$$\begin{cases} -\mathrm{i}kC_0 u_i - \dfrac{\mathrm{i}k}{k_\varepsilon^2}\displaystyle\sum_{\ell=1}^{N_p} A_\ell \Delta_{\Sigma_{ij}} \varphi_{i,\ell} = -\mathrm{i}kC_0 u_j - \dfrac{\mathrm{i}k}{k_\varepsilon^2}\displaystyle\sum_{\ell=1}^{N_p} A_\ell \Delta_{\Sigma_{ji}} \varphi_{j,\ell} \\ \partial_{\mathbf{n}_i} u_i = -\partial_{\mathbf{n}_j} u_j \end{cases}, \qquad \text{on } \Sigma_{ij}, \quad \text{(A.47)}$$

where $\varphi_{i,\ell}$ satisfy $\left(1 + \frac{1}{k_\varepsilon^2} B_\ell \Delta_{\Sigma_{ij}}\right) \varphi_{i,\ell} = u_i, \forall \ell = 1, \cdots, N_p$. Just as before, we will rewrite this as transmission conditions to be used as boundary conditions for the local BVPs. By adding and subtracting the equations in (A.47) we obtain:

$$\begin{cases} -\mathrm{i}kC_0 u_i - \dfrac{\mathrm{i}k}{k_\varepsilon^2}\displaystyle\sum_{\ell=1}^{N_p} A_\ell \Delta_{\Sigma_{ij}} \varphi_{i,\ell} + \partial_{\mathbf{n}_i} u_i = \underbrace{-\mathrm{i}kC_0 u_j - \dfrac{\mathrm{i}k}{k_\varepsilon^2}\displaystyle\sum_{\ell=1}^{N_p} A_\ell \Delta_{\Sigma_{ji}} \varphi_{j,\ell} - \partial_{\mathbf{n}_j} u_j}_{\text{define } g_{ij}(u_j)} \\ \underbrace{-\mathrm{i}kC_0 u_i - \dfrac{\mathrm{i}k}{k_\varepsilon^2}\displaystyle\sum_{\ell=1}^{N_p} A_\ell \Delta_{\Sigma_{ij}} \varphi_{i,\ell} - \partial_{\mathbf{n}_i} u_i}_{\text{define } g_{ji}(u_i)} = -\mathrm{i}kC_0 u_j - \dfrac{\mathrm{i}k}{k_\varepsilon^2}\displaystyle\sum_{\ell=1}^{N_p} A_\ell \Delta_{\Sigma_{ji}} \varphi_{j,\ell} + \partial_{\mathbf{n}_j} u_j \end{cases} \quad \text{(A.48)}$$

on the interface Σ_{ij}. Now we can express the local BVP on domain Ω_i:

$$\begin{cases} \Delta u_i + k^2 u_i = 0 & x \in \Omega_i \\ \beta u_i + \partial_{\mathbf{n}_i} u_i = -\beta u^{\text{inc}} - \partial_{\mathbf{n}} u^{\text{inc}} & x \in \Gamma_i \\ -\mathrm{i}kC_0 u_i - \dfrac{\mathrm{i}k}{k_\varepsilon^2}\displaystyle\sum_{\ell=1}^{N_p} A_\ell \Delta_{\Sigma_{ij}} \varphi_{i,\ell} + \partial_{\mathbf{n}_i} u_i = g_{ij} & x \in \Sigma_{ij} : j \in \Lambda_i, \\ \left(1 + \dfrac{1}{k_\varepsilon^2} B_\ell \Delta_{\Sigma_{ij}}\right) \varphi_{i,\ell} = u_i, & \forall \ell = 1, \cdots, N_p. \end{cases} \quad \text{(A.49)}$$

As before, augment Λ_i to include 0 and set $g_{i0} = 0$. To obtain the weak form, multiply by a test function $v_i \in H_0^1$, integrate, and use integration by parts. The calculation

for the third integral in (A.49) is

$$\begin{aligned}\int_{\Sigma_{ij}} \partial_{\mathbf{n}_i} u_i v_i \,\mathrm{d}s &= \int_{\Sigma_{ij}} g_{ij} v_i \,\mathrm{d}s + \int_{\Sigma_{ij}} \mathrm{i}kC_0 u_i v_i \,\mathrm{d}s + \int_{\Sigma_{ij}} \frac{\mathrm{i}k}{k_\varepsilon^2} \sum_{\ell=1}^{N_p} A_\ell \Delta_{\Sigma_{ij}} \varphi_{i,\ell} v_i \,\mathrm{d}s \\ &= \int_{\Sigma_{ij}} g_{ij} v_i \,\mathrm{d}s + \mathrm{i}kC_0 \int_{\Sigma_{ij}} u_i v_i \,\mathrm{d}s + \sum_{\ell=1}^{N_p} A_\ell \frac{\mathrm{i}k}{k_\varepsilon^2} \int_{\Sigma_{ij}} \Delta_{\Sigma_{ij}} \varphi_{i,\ell} v_i \,\mathrm{d}s \\ &= \int_{\Sigma_{ij}} g_{ij} v_i \,\mathrm{d}s + \mathrm{i}kC_0 \int_{\Sigma_{ij}} u_i v_i \,\mathrm{d}s - \sum_{\ell=1}^{N_p} A_\ell \frac{\mathrm{i}k}{k_\varepsilon^2} \int_{\Sigma_{ij}} \nabla_{\Sigma_{ij}} \varphi_{i,\ell} \cdot \nabla_{\Sigma_{ij}} v_i \,\mathrm{d}s.\end{aligned} \tag{A.50}$$

For the fourth integral in (A.49), the calculation is

$$\begin{aligned}0 &= \int_{\Sigma_{ij}} \left(\left(1 + \frac{1}{k_\varepsilon^2} B_\ell \Delta_{\Sigma_{ij}} \right) \varphi_{i,\ell} - u_i \right) v_i \mathrm{d}s \\ &= -\int_{\Sigma_{ij}} u_i v_i \mathrm{d}s + \int_{\Sigma_{ij}} \varphi_{i,\ell} v_i \mathrm{d}s + \frac{B_\ell}{k_\varepsilon^2} \int_{\Sigma_{ij}} \Delta_{\Sigma_{ij}} \varphi_{i,\ell} \, v_i \mathrm{d}s \\ &= -\int_{\Sigma_{ij}} u_i v_i \mathrm{d}s + \int_{\Sigma_{ij}} \varphi_{i,\ell} v_i \mathrm{d}s - \frac{B_\ell}{k_\varepsilon^2} \int_{\Sigma_{ij}} \nabla_{\Sigma_{ij}} \varphi_{i,\ell} \cdot \nabla_{\Sigma_{ij}} v_i \mathrm{d}s\end{aligned} \tag{A.51}$$

If we put it altogether, our bilinear system becomes

$$\begin{cases} a_i(u_i, v_i) + \displaystyle\sum_{\ell=1}^{N_p} p_i(\phi_{i,\ell}, v_i) = g_i(v_i) \\ \\ q_i(u_i, v_i) + r_i(\phi_{i,\ell}, v_i) = 0 \qquad\qquad \forall \ell = 1, \cdots, N_p, \end{cases} \tag{A.52}$$

where the bilinear/linear operators are given by

$$\begin{aligned} a_i(u_i, v_i) &= \int_{\Omega_i} (\nabla u_i \cdot \nabla v_i - k^2 u_i v_i) \mathrm{d}x + \int_{\Gamma_i} \beta u_i \, v_i \,\mathrm{d}s + \sum_{j \in \Lambda_i} -\mathrm{i}kC_0 \int_{\Sigma_{ij}} u_i v_i \mathrm{d}s \\ p_i(\phi_{i,\ell}, v_i) &= \sum_{j \in \Lambda_i} A_\ell \frac{\mathrm{i}k}{k_\varepsilon^2} \int_{\Sigma_{ij}} \nabla_{\Sigma_{ij}} \varphi_{i,\ell} \cdot \nabla_{\Sigma_{ij}} v_i \,\mathrm{d}s \\ g_i(v_i) &= -\int_{\Gamma_i} (\beta u^{\mathrm{inc}} + \partial_{\mathbf{n}_i} u^{\mathrm{inc}}) \, v_i \,\mathrm{d}s + \sum_{j \in \Lambda_i} \int_{\Sigma_{ij}} g_{ij} v_i \,\mathrm{d}s \\ q_i(u_i, v_i) &= \sum_{j \in \Lambda_i} \int_{\Sigma_{ij}} u_i v_i \,\mathrm{d}s \\ r_i(\phi_{i,\ell}, v_i) &= \sum_{j \in \Lambda_i} -\int_{\Sigma_{ij}} \varphi_{i,\ell} v_i \,\mathrm{d}s + \frac{B_\ell}{k_\varepsilon^2} \int_{\Sigma_{ij}} \nabla_{\Sigma_{ij}} \varphi_{i,\ell} \cdot \nabla_{\Sigma_{ij}} v_i \mathrm{d}s. \end{aligned} \tag{A.53}$$

To derive the update formula, we consider two domains Ω_i and Ω_j with a common Σ_{ij}. Each domain has its own local BVP for which we impose the iterative boundary conditions:

$$\begin{aligned} -\mathrm{i}kC_0 u_i^{(n)} - \frac{\mathrm{i}k}{k_\varepsilon^2}\sum_{\ell=1}^{N_p} A_\ell \Delta_{\Sigma_{ij}} \varphi_{i,\ell}^{(n)} + \partial_{\mathbf{n}_i} u_i^{(n)} &= g_{ij}^{(n-1)} \qquad \text{for domain } \Omega_i, \\ -\mathrm{i}kC_0 u_j^{(n)} - \frac{\mathrm{i}k}{k_\varepsilon^2}\sum_{\ell=1}^{N_p} A_\ell \Delta_{\Sigma_{ji}} \varphi_{j,\ell}^{(n)} + \partial_{\mathbf{n}_j} u_j^{(n)} &= g_{ji}^{(n-1)} \qquad \text{for domain } \Omega_j. \end{aligned} \tag{A.54}$$

To be explicit, we define $\varphi_{m,\ell}$ to be the function satisfying $\left(1 + \frac{1}{k_\varepsilon^2} B_\ell \Delta_{\Sigma_{mj}}\right) \varphi_{m,\ell} = u_m$ for all ℓ, and for $m = i, j$. And we define $g_{ij}^{(n)}$ and $g_{ji}^{(n)}$ to be the auxiliary unknowns defined in (A.48) evaluated using the arguments $u_j^{(n)}$ and $u_i^{(n)}$ respectively. At each iteration, we will simultaneously solve for $u_i^{(n)}$ and $\varphi_{j,\ell}^{(n)}$ for all ℓ using the boundary data $g_{ij}^{(n-1)}$ for all j.Then we use following update formula to increment the boundary data:

$$g_{ij}^{(n)} = -2\mathrm{i}kC_0 u_j^{(n)} - \frac{2\mathrm{i}k}{k_\varepsilon^2}\sum_{\ell=1}^{N_p} A_\ell \Delta_{\Sigma_{ij}} \varphi_{j,\ell}^{(n)} - g_{ji}^{(n-1)} \tag{A.55}$$

We must multiply the update formula with a test function $v_i \in H_0^1$ then do integration by parts to obtain the actual update formula used in the NDDM-FEM iteration:

$$\begin{aligned} \int_{\Sigma_{ij}} g_{ij}^{(n)} v_i \,\mathrm{d}s = -2\mathrm{i}kC_0 \int_{\Sigma_{ij}} u_j^{(n)} v_i \,\mathrm{d}s + \\ \frac{2\mathrm{i}k}{k_\varepsilon^2}\sum_{\ell=1}^{N_p} A_\ell \int_{\Sigma_{ij}} \nabla_{\Sigma_{ij}} \varphi_{j,\ell}^{(n)} \cdot \nabla_{\Sigma_{ij}} v_i \,\mathrm{d}s - \int_{\Sigma_{ij}} g_{ji}^{(n-1)} v_i \,\mathrm{d}s. \end{aligned} \tag{A.56}$$

APPENDIX B

MONGE-AMPÈRE

Appendix B is devoted to additional information regarding the domain decomposition methods for the Monge-Ampère equation.

B.1 PETSc's Cartesian Mesh

This section describes PETSc's interface for nonlinear PDEs on Cartesian meshes. PETSc provides nonlinear solvers, linear solvers, preconditioners, and options for combining them. Both structured and unstructured grids are supported. On a Cartesian mesh, the user supplies local residual evaluations, while PETSc manages the parallel data distribution and communication.

Rectangular domains allow direct control of mesh dimensions and subdomain layouts. The Cartesian structure also simplifies indexing and stencil evaluation. The following example explains how to supply the residual needed by PETSc's parallel nonlinear solvers.

Let us discuss how the Cartesian implementation works in the case of the Monge-Ampère equation. The continuous version of our nonlinear operator is

$$F(u(x)) = \begin{cases} -\det{}^{+}(D^2u(x)) + f(x) & \text{if } x \in \Omega \subset \mathbb{R}^2 \\ u(x) - g(x) & \text{if } x \in \partial\Omega \end{cases} \tag{B.1}$$

where f, g, and $\det^{+}$ are as defined in Subsection 4.1.1. Note that F is also called the residual function. Let the domain Ω be a square $[-L, L]^2$ meshed by a lattice of $N \times N$ nodes equally spaced apart. The mesh coordinates are given by $x_i = -L + ih$ and $y_j = -L + jh$, where the step size is $h = \frac{2L}{N+1}$. We denote our numerical approximation to $F(u) = 0$ as

$$(\mathbf{u})_{i,j} := u_{i,j} \approx u(x_i, y_j) \tag{B.2}$$

for $1 \leq i, j \leq N$. The variable $\mathbf{u}$ is an N by N real-valued matrix, which we may also express as `u[N][N]`. Let F^h denote the discrete form of Equation (B.1). It can be any wide stencil scheme such as the determinant integral representation quadrature Equation (4.18), for example. We define the vector form of F as

$$(\mathbf{F})_{i,j} := F_{i,j} \approx F(u(x_i, y_j)), \tag{B.3}$$

We use `F[i][j]` for entry (i, j) of the $N \times N$ array `F[N][N]`. PETSc requires a routine that takes $\mathbf{u}$ and computes $\mathbf{F}$, evaluating each $F_{i,j}$ from $u_{i,j}$ and neighboring stencil values. **At minimum, a residual evaluation routine must be supplied to use the nonlinear solvers.**

We will now describe how to code the residual function for the wide stencil scheme discussed in Subsection 4.2.3. Let $\mathbf{u}_{ij}$ denote the vector containing the evaluations of u at the stencil points which contributes to the value of $F_{i,j}$. For a given integer stencil width d, there are $4d + 1$ points including the center. Part of the algorithm is to loop around the angular discretization and fetch all of the stencil points which contribute to the value of the Monge-Ampère operator. The points are given by $u_{i+di,j+dj}$ where $|di| + |dj| = d$. If any of these points lie outside of the boundary, they must be projected onto the boundary. They must be projected in a way that preserves the reference angle of the stencil point. This will be discussed in more detail in a later section. Once $\mathbf{u}_{ij}$ has been fetched, it is then used to approximate the Monge-Ampère operator which we denote as

$$\tilde{\Theta}_{ij}(\mathbf{u}_{ij}) \approx \det{}^{+}(D^2 u(x_i, y_j)). \tag{B.4}$$

This scalar quantity can be expressed as

$$\det{}_{\varepsilon}^{+}(D^2 u_{i,j}) \approx \pi^2 \left[\sum_{k=0}^{2d} \frac{w_k}{\max(\mathcal{D}_{\theta_k}^2 u_{i,j}, \varepsilon)} \right]^{-2} + \min_k \{\min(\mathcal{D}_{\theta_k}^2 u_{i,j}, \varepsilon)\}. \tag{B.5}$$

The repeated quantity $\mathcal{D}^2_{\theta_k} u_{i,j}$ in Equation (B.5) need be computed only once for each k. Store the values for $k = 1$ to $M := 2d$ in `SDD[M]`, an array of second directional derivatives. Once these values are available, evaluating $\tilde{\Theta}_{ij}$ is straightforward. The quadrature weights w_k are computed once before iteration.

Algorithm 12: Method to compute the residual function for the 2D Monge-Ampère equation using a wide stencil scheme

Data: `u[N][N]`
Result: `F[N][N]`
for $i = 1 : N$ **do**
  for $j = 1 : N$ **do**
    Compute `SDD[M]` for `u[i][j]`;
    Compute $\tilde{\Theta}_{ij}$ which depends on `SDD` ;
    `F[i][j]` $= -\tilde{\Theta}_{ij} + f(x_i, y_j)$;
  end
end

Algorithm 12 maps $\mathbf{u}$ to a residual array $\mathbf{F}$ with the same size and ordering. It loops over Cartesian nodes (x_i, y_j) and stores the corresponding residual in `F[i][j]`. Neither `F[N][N]` nor `u[N][N]` needs to be manually flattened to a vector of size $N^2 \times 1$. PETSc handles the mapping from grid indices to algebraic unknowns. The same approach extends to 3D; evaluating `SDD` remains the main implementation task.

Once the residual is implemented, PETSc can approximate its Jacobian by finite differences, with coloring to reduce the evaluation cost. The wide stencil requires a box-type communication pattern because it includes diagonal grid points. PETSc then distributes the computation over the requested subdomains. The experiments in this dissertation instead use an explicitly coded Jacobian, as noted below.

B.2 Computing the Second Directional Derivative

Computing the SDDs is difficult for two reasons. First, it must be generalized for any stencil width $d \geq 1$. Second, the stencil points may need to be projected if they lie outside of the computational domain. The SDD for $u(x_i, y_j)$ on the Cartesian grid

for a general direction ν_k is given by:

$$\mathcal{D}^2_{\nu_k} u(x_i, y_j) \approx \\ 2\frac{h_k^- u(x_i + hS_{i,k}, y_j + hS_{j,k}) + h_k^+ u(x_i - hS_{i,k}, y_j - hS_{j,k}) - (h_k^+ + h_k^-)u(x_i, y_j)}{h_k^+ h_k^- (h_k^+ + h_k^-)}, \tag{B.6}$$

where S is the integer amount of nodes to move in either the x or y direction. The algorithm to compute the SDDs for the kth direction is given in Algorithm 13.

Algorithm 13: Algorithm for boundary free mesh of arbitrary width.

Data: $x, y, N_x, N_y, i, j, k, h$
Result: Second directional derivative `SDD[k]`
for *Dir = [Fwd Bak]* **do**
  $(S_i, S_j) = Stencil_{Dir}[k]$
  if $(i, j) + (S_i, S_j) \notin$ *Interior* **then**
    $(\Delta i, \Delta j) =$ Algorithm14$(N_x, N_y, i, j, S_i, S_j)$
    $u_{Dir}[k] = g(x + h\Delta i, y + h\Delta j)$
    $h_{Dir}[k] = h\sqrt{\Delta i^2 + \Delta j^2}$
  else
    $(\Delta i, \Delta j) = (S_i, S_j)$
    $u_{Dir}[k] = u[j + \Delta j][i + \Delta i]$
    $h_{Dir}[k] = h\sqrt{\Delta i^2 + \Delta j^2}$
  end
end
`SDD[k]` $= 2\frac{h_{Bak}[k]u_{Fwd}[k] + h_{Fwd}[k]u_{Bak}[k] - (h_{Bak} + h_{Fwd}[k])u[j][i]}{h_{Fwd}[k]h_{Bak}[k](h_{Fwd}[k] + h_{Bak}[k]}$)

Algorithm 13 assumes a general Cartesian mesh of size $N_x \times N_y$ and the direction ν_k has already been computed. The line which reads $(S_i, S_j) = Stincil_{Dir}[k]$ means the direction of the derivatives are pre-computed and known. The values S_i and S_j are integers which correspond to the number of nodes displaced in the x or y direction with respect to the reference node, when no projection is needed. In general, the values S_i and S_j need to be adjusted if they are near the boundary. Algorithm 14 computes the new projection pair $(\Delta i, \Delta j)$ which may not be integers.

Algorithm 14 only runs if the stencil point falls outside of the computational domain. When it does, the algorithm computes new fractional steps such that the reference angle between (x_i, y_j) and $(x_i + h_x S_i, y_j + h_y S_j)$ is preserved and lies on $\partial\Omega$. The notation $v = (check)?v_1 : v_2$ means that the value assigned to v depends on the Boolean value of *check*. If *check* is true $v = v_1$. If *check* is false, then $v = v_2$.

Algorithm 14: Algorithm for finding projection index when near boundary.

Data: N_x, N_y, i, j, S_i, S_j
Result: Projection Stencil $(\Delta i, \Delta j)$
if $S_i == 0$ **then**
  $\Delta i = 0$
  $\Delta j = (S_j > 0) \quad ? \quad N_y - j : -(1 + j)$
else if $S_j == 0$ **then**
  $\Delta i = (S_i > 0) \quad ? \quad N_x - i : -(1 + i)$
  $\Delta j = 0$
else
  $m = S_j / S_i$
  if $S_i > 0 \,\&\, S_j > 0$ **then**
    $check = |\frac{N_y - j}{m}| < |N_x - i|$
    $\Delta i = (check) \quad ? \quad \frac{N_y - j}{m} : N_x - i$
    $\Delta j = (check) \quad ? \quad N_y - j : m(N_x - i)$
  else if $S_i > 0 \,\&\, S_j < 0$ **then**
    $check = |N_x - i| < |\frac{1+j}{m}|$
    $\Delta i = (check) \quad ? \quad N_x - i : -\frac{1+j}{m}$
    $\Delta j = (check) \quad ? \quad m(N_x - i) : -(1 + j)$
  else if $S_i < 0 \,\&\, S_j < 0$ **then**
    $check = |\frac{1+j}{m}| < |1 + i|$
    $\Delta i = (check) \quad ? \quad -\frac{1+j}{m} : -(1 + i)$
    $\Delta j = (check) \quad ? \quad -(1 + j) : -m(1 + i)$
  else
    $check = |1 + i| < |\frac{N_y - j}{m}|$
    $\Delta i = (check) \quad ? \quad -(1 + i) : \frac{N_y - j}{m}$
    $\Delta j = (check) \quad ? \quad -m(1 + i) : N_y - j$
  end
end

PETSc also accepts an explicit Jacobian routine, which can be faster than finite difference approximation. All numerical experiments in this dissertation use a hand-coded Jacobian.

REFERENCES

[1] Alfonzetti, S., Borzì, G., and Salerno, N. Iteratively-improved Robin Boundary Conditions for the Finite Flement Solution of Scattering Problems in Unbounded Domains. *International Journal for Numerical Methods in Engineering 42*, 4 (June 1998), 601–629.

[2] Antoine, X. Fast approximate computation of a time-harmonic scattered field using the on-surface radiation condition method. *IMA Journal of Applied Mathematics 66*, 1 (Jan. 2001), 83–110.

[3] Antoine, X., Barucq, H., and Bendali, A. Bayliss–Turkel-like Radiation Conditions on Surfaces of Arbitrary Shape. *Journal of Mathematical Analysis and Applications 229*, 1 (Jan. 1999), 184–211.

[4] Balay, S., Abhyankar, S., Adams, M. F., Benson, S., Brown, J., Brune, P., Buschelman, K., Constantinescu, E., Dalcin, L., Dener, A., Eijkhout, V., Faibussowitsch, J., Gropp, W. D., Hapla, V., Isaac, T., Jolivet, P., Karpeev, D., Kaushik, D., Knepley, M. G., Kong, F., Kruger, S., May, D. A., McInnes, L. C., Mills, R. T., Mitchell, L., Munson, T., Roman, J. E., Rupp, K., Sanan, P., Sarich, J., Smith, B. F., Zampini, S., Zhang, H., Zhang, H., and Zhang, J. PETSc/TAO users manual. Tech. Rep. ANL-21/39 - Revision 3.18, Argonne National Laboratory, Lemont, IL, 2022.

[5] Balay, S., Abhyankar, S., Adams, M. F., Benson, S., Brown, J., Brune, P., Buschelman, K., Constantinescu, E. M., Dalcin, L., Dener, A., Eijkhout, V., Faibussowitsch, J., Gropp, W. D., Hapla, V., Isaac, T., Jolivet, P., Karpeev, D., Kaushik, D., Knepley, M. G., Kong, F., Kruger, S., May, D. A., McInnes, L. C., Mills, R. T., Mitchell, L., Munson, T., Roman, J. E., Rupp, K., Sanan, P., Sarich, J., Smith, B. F., Zampini, S., Zhang, H., Zhang, H., and Zhang, J. Portable, Extensible Toolkit for Scientific Computation Web page. https://petsc.org/, Retrieved on 2023-04-14.

[6] Bardi, M., and Mannucci, P. Comparison principles and Dirichlet Problem for Fully Nonlinear Degenerate Equations of Monge–Ampère type. *Forum Mathematicum 25*, 6 (Nov. 2013), 1291–1330.

[7] Barles, G., and Souganidis, P. E. Convergence of Approximation Schemes for Fully Nonlinear Second order Equations. *Asymptotic Analysis 4*, 3 (1991), 271–283.

[8] Bayliss, A., Gunzburger, M., and Turkel, E. Boundary Conditions for the Numerical Solution of Elliptic Equations in Exterior Regions. *SIAM Journal on Applied Mathematics 42*, 2 (Apr. 1982), 430–451.

[9] Benamou, J.-D., Collino, F., and Mirebeau, J.-M. Monotone and Consistent Discretization of the Monge-Ampère operator. *Mathematics of Computation 85*, 302 (2016), 2743–2775.

[10] Bendali, A., and Boubendir, Y. Non-overlapping Domain Decomposition Method for a Nodal Finite Element Method. *Numerische Mathematik 103*, 4 (June 2006), 515–537.

[11] Bendali, A., Boubendir, Y., and Zerbib, N. Localized Adaptive Radiation Condition for Coupling Boundary and Finite Element Methods Applied to Wave Propagation Problems. *Institute of Mathematics and its Applications Journal of Numerical Analysis 34*, 3 (July 2014), 1240–1265.

[12] Bendali, A., and Fares, M. Boundary Integral Equations Methods in Acoustic Scattering. In *Computational Science, Engineering & Technology Series*, vol. 18. Saxe-Coburg Publications, Stirlingshire, UK, Aug. 2008, pp. 1–36.

[13] Berenger, J. P. A Perfectly Matched Layer for the Absorption of Electromagnetic Waves. *Journal of Computational Physics 114*, 2 (Oct. 1994), 185–200.

[14] Boubendir, Y. *Techniques de Décomposition de Domaine et Méthodes D'équations Intégrales*. PhD thesis, National Institute of Applied Sciences of Toulouse, Toulouse, France, 2002.

[15] Boubendir, Y. An Analysis of the BEM-FEM Non-Overlapping Domain Decomposition Method for a Scattering Problem. *Journal of Computational and Applied Mathematics 204*, 2 (July 2007), 282–291.

[16] Boubendir, Y., Antoine, X., and Geuzaine, C. A Quasi-optimal Non-overlapping Domain Decomposition Algorithm for the Helmholtz Equation. *Journal of Computational Physics 231*, 2 (Jan. 2012), 262–280.

[17] Boubendir, Y., Bendali, A., and Fares, M. Coupling of a Non-Overlapping Domain Decomposition Method for a Nodal Finite Element Method with a Boundary Element Method. *International Journal for Numerical Methods in Engineering 73*, 11 (Mar. 2008), 1624–1650.

[18] Boubendir, Y., Jerez-Hanckes, C., Pérez-Arancibia, C., and Turc, C. Domain Decomposition Methods Based on Quasi-Optimal Transmission Operators for the Solution of Helmholtz Transmission Problems. *ArXiv* (Oct. 2017). arXiv: 1710.02694.

[19] Boubendir, Y., and Midura, D. Non-Overlapping Domain Decomposition Algorithm Based on Modified Transmission Conditions for the Helmholtz Equation. *Computers & Mathematics with Applications 75*, 6 (Mar. 2018), 1900–1911.

[20] Brune, P. R., Knepley, M. G., Smith, B. F., and Tu, X. Composing Scalable Nonlinear Algebraic Solvers. *SIAM Review 57*, 4 (Jan. 2015), 535–565.

[21] Bruno, O., Elling, T., Paffenroth, R., and Turc, C. Electromagnetic Integral Equations Requiring Small Numbers of Krylov-Subspace Iterations. *Journal of Computational Physics 228*, 17 (2009), 6169–6183.

[22] Brusca, J., and Hamfeldt-Froese, B. D. A Convergent Quadrature Based Method for the Monge-Ampère Equation. *ArXiv* (May 2022). arXiv:2205.03483.

[23] Budd, C. J., and Williams, J. F. Moving Mesh Generation using the Parabolic Monge-Ampère Equation. *SIAM Journal on Scientific Computing* (2008).

[24] Caffarelli, L., Nirenberg, L., and Spruck, J. The Dirichlet Problem for Nonlinear Second-order Elliptic Equations I. Monge-Ampére Equation. *Communications on Pure and Applied Mathematics 37*, 3 (May 1984), 369–402.

[25] Cai, X.-C., Gropp, W. D., Keyes, D. E., and Tidriri, M. D. Newton-Krylov-Schwarz Methods in CFD. In *Numerical methods for the Navier-Stokes equations*. Vieweg+Teubner Verlag, Wiesbaden, 1994, pp. 17–30.

[26] Cai, X.-C., Keyes, D. E., and Marcinkowski, L. Non-linear Additive Schwarz Preconditioners and Application in Computational Fluid Dynamics. *International Journal for Numerical Methods in Fluids 40*, 12 (Dec. 2002), 1463–1470.

[27] Chan, T. F., and VanderVorst, H. A. Approximate and Incomplete Factorizations. In *Parallel Numerical Algorithms*, M. D. Salas, D. E. Keyes, A. Sameh, and V. Venkatakrishnan, Eds., vol. 4. Springer Netherlands, Dordrecht, 1997, pp. 167–202. Series Title: ICASE/LaRC Interdisciplinary Series in Science and Engineering.

[28] Chen, Y., Wan, J. W. L., and Lin, J. Monotone Mixed Finite Difference Scheme for Monge–Ampère Equation. *Journal of Scientific Computing 76*, 3 (2018), 1839–1867.

[29] Crandall, M. G., Ishii, H., and Lions, P. L. User's Guide to Viscosity Solutions of Second order Partial Differential Equations. *Bulletin of the American Mathematical Society* (1992).

[30] Darbas, M. Generalized Combined Field Integral Equations for the Iterative Solution of the Three-Dimensional Maxwell Equations. *Applied Mathematics Letters 19*, 8 (2006), 834–839.

[31] Datta, B. N. *Numerical Linear Algebra and Applications*, 2nd ed ed. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2010.

[32] Desprès, B. *Méthodes de Décomposition de Domaine pour les Problemes de Propagation Dóndes en Régime Harmonique*. PhD thesis, University of Paris, Paris, France, 1991.

[33] Dolean, V., Gander, M. J., Kheriji, W., Kwok, F., and Masson, R. Nonlinear Preconditioning: How to Use a Nonlinear Schwarz Method to Precondition Newton's Method. *SIAM Journal on Scientific Computing 38*, 6 (Jan. 2016), A3357–A3380.

[34] Eijkhout, V. *Introduction to High-Performance Scientific Computing*, 2nd edition, revision 2015 ed. Lulu, Morrisville, NC, 2015.

[35] Eisenstat, S. C. Efficient Implementation of a Class of Preconditioned Conjugate Gradient Methods. *SIAM Journal on Scientific and Statistical Computing 2*, 1 (Mar. 1981), 1–4.

[36] Engquist, B., and Majda, A. Absorbing Boundary Conditions for the Numerical Simulation of Waves. *Mathematics of Computation 31*, 139 (1977), 629–651.

[37] Erhel, J., Burrage, K., and Pohl, B. Restarted GMRES preconditioned by deflation. *Journal of Computational and Applied Mathematics 69*, 2 (May 1996), 303–318.

[38] Feng, Z., Froese, B. D., Huang, C., Ma, D., and Liang, R. Creating Unconventional Geometric Beams with Large Depth of Field using Double Freeform-surface Optics. *Applied Optics 54*, 20 (July 2015), 6277.

[39] Froese, B. D. Convergent Approximation of Non-continuous Surfaces of Prescribed Gaussian Curvature. *ArXiv* (2016). 10.48550/ARXIV.1601.06315.

[40] Froese, B. D. Meshfree Finite Difference Approximations for Functions of the Eigenvalues of the Hessian. *Numerische Mathematik 138*, 1 (Jan. 2018), 75–99.

[41] Froese, B. D., and Oberman, A. M. Convergent Finite Difference Solvers for Viscosity Solutions of the Elliptic Monge-Ampère Equation in Dimensions Two and Higher. *ArXiv* (2010).

[42] Gander, M. J., Magoulès, F., and Nataf, F. Optimized Schwarz Methods without Overlap for the Helmholtz Equation. *SIAM Journal on Scientific Computing 24*, 1 (Jan. 2002), 38–60.

[43] Gander, M. J., and Wanner, G. The Origins of the Alternating Schwarz Method. In *Domain Decomposition Methods in Science and Engineering XXI*, J. Erhel, M. J. Gander, L. Halpern, G. Pichot, T. Sassi, and O. Widlund, Eds., vol. 98. Springer International Publishing, Cham, 2014, pp. 487–495.

[44] Gockenbach, M. S. *Understanding and Implementing the Finite Element Method*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2006.

[45] Hamfeldt, B. F. Convergence Framework for the Second Boundary Value Problem for the Monge–Ampère Equation. *SIAM Journal on Numerical Analysis 57*, 2 (Jan. 2019), 945–971.

[46] Heitmann, K., Frontiere, N., Sewell, C., Habib, S., Pope, A., Finkel, H., Rizzi, S., Insley, J., and Bhattacharya, S. The q continuum simulation: Harnessing the power of gpu accelerated supercomputers. *The Astrophysical Journal Supplement Series 219*, 2 (Aug. 2015), 34.

[47] Jackson, J. D. *Classical Electrodynamics*, 3rd ed. Wiley, New York, NY, 1999.

[48] Jian, H., and Tu, X. The Oblique Boundary Value Problem for Monge-Ampère Equations. *ArXiv* (2021).

[49] Lahaye, D., Tang, J., and Vuik, K., Eds. *Modern Solvers for Helmholtz Problems*. Geosystems Mathematics. Springer International Publishing, Cham, CH, 2017.

[50] Lawrence, S. Developable Surfaces: Their History and Application. *Nexus Network Journal 13*, 3 (Oct. 2011), 701–714.

[51] Lions, P. L. On the Schwarz Alternating Method III: A Variant for Nonoverlapping Subdomains. In *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations* (1990), Society for Industrial and Applied Mathematics, pp. 202–223.

[52] M. Oberman, A. Wide Stencil Finite Difference Schemes for the Elliptic Monge-Ampère Equation and Functions of the Eigenvalues of the Hessian. *Discrete & Continuous Dynamical Systems - B 10*, 1 (2008), 221–238.

[53] Midura, D. *Efficient Domain Decomposition Algorithms for the Solution of the Helmholtz Equation*. PhD thesis, New Jersey Institute of Technology, Newark, NJ, July 2014.

[54] Milinazzo, F. A., Zala, C. A., and Brooke, G. H. Rational Square-Root Approximations for Parabolic Equation Algorithms. *The Journal of the Acoustical Society of America 101*, 2 (Feb. 1997), 760–766.

[55] Moré, J. J., and Sorensen, D. C. Computing a Trust Region Step. *SIAM Journal on Scientific and Statistical Computing 4*, 3 (Sept. 1983), 553–572.

[56] Motzkin, T. S., and Wasow, W. On the Approximation of Linear Elliptic Differential Equations by Difference Equations with Positive Coefficients. *Journal of Mathematics and Physics 31*, 1-4 (Apr. 1952), 253–259.

[57] Oberman, A. M. Convergent Difference Schemes for Degenerate Elliptic and Parabolic Equations: Hamilton–Jacobi Equations and Free Boundary Problems. *SIAM Journal on Numerical Analysis 44*, 2 (Jan. 2006), 879–895.

[58] Peyre, G., and Cuturi, M. Computational optimal transport. *Foundations and Trends in Machine Learning 11*, 5-6 (2019), 355–607.

[59] Saad, Y. *Iterative Methods for Sparse Linear Systems*, 2nd ed ed. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2003.

[60] Sequeira, A., and Nedelec, J. C. The Coupling of Boundary Integral and Finite Element Methods for the Bidimensional Exterior Steady Stokes Problem. *Mathematical Methods in the Applied Sciences 5*, 1 (1983), 356–375.

[61] St-Cyr, A., Gander, M. J., and Thomas, S. J. Optimized Multiplicative, Additive, and Restricted Additive Schwarz Preconditioning. *SIAM Journal on Scientific Computing 29*, 6 (Jan. 2007), 2402–2425.

[62] Steinbach, O. Boundary Integral Equations for Helmholtz Boundary Value and Transmission Problems. In *Direct and Inverse Problems in Wave Propagation and Applications*, I. Graham, U. Langer, J. Melenk, and M. Sini, Eds. De Gruyter, 2013, pp. 253–292.

[63] Toselli, A., and Widlund, O. B. *Domain Decomposition Methods - Algorithms and Theory*. No. 34 in Springer Series in Computational Mathematics. Springer, Berlin, Germany, 2005.

[64] Villani, C. *Optimal Transport: Old and New*. No. 338 in Grundlehren der mathematischen Wissenschaften. Springer, Berlin, Germany, 2009. OCLC: ocn244421231.

[65] Wilcox, C. H. Scattering Theory for the d'alembert Equation in Exterior Domains. *Lecture Notes in Mathematics* (1975).

[66] Yamazaki, I., Hoemmen, M., Luszczek, P., and Dongarra, J. Improving Performance of GMRES by Reducing Communication and Pipelining Global Collectives. In *2017 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)* (Buena Vista, FL, May 2017), IEEE, pp. 1118–1127.