Sparse Recovery of Poincaré-Steklov Operators

A Thesis
Presented to
The Academic Faculty

by

Christian Engman

In Partial Fulfillment
of the Requirements for the Degree
Bachelors of Science in Computer Science in the
College of Computing

Georgia Institute of Technology
December 2025

Sparse Recovery of Poincaré-Steklov Operators

Approved by:

Dr. Florian Schäfer, Advisor
School of Computational Science and Engineering
Georgia Institute of Technology

Dr. Qi Tang
School of Computational Science and Engineering
Georgia Institute of Technology

Date Approved: December 3, 2025

Abstract

Poincare-Steklov operators arise from elliptic partial differential equations, representing transformations between different types of boundary conditions. Poincare-Steklov operators and their discretizations are similar in structure to integral and differential operators, and the Neumann-to-Dirichlet map in particular resembles an elliptic solution operator. Previous work by Schäfer and Owhadi [11] has shown that the Cholesky factors of elliptic solvers in d spatial dimensions can be approximated to accuracy ε in $\mathcal{O}(\log(N)\log^d(N/\varepsilon))$ matrix vector products. In this work, we give an algorithm that aims to similarly approximate the Cholesky factor of a Neumann-to-Dirichlet operator to ε accuracy in a logarithmic number matrix-vector products with the operator, accessed only as a black box using carefully chosen right-hand sides. We verify this cost-accuracy tradeoff empirically on several different problems. Additionally, we have found that our approximation can be applied to inverse problems arising from elliptic PDEs. Poincare-Steklov operators appear naturally in elliptic inverse problems, as they model information that can be measured only from the boundary of a physical domain. We have seen promising results when using our approximation as a stand-in for the exact operator when solving these inverse problems, typically with better performance compared to an optimal low-rank approximation derived from an equal number of matrix-vector products.

Contents

1 Introduction

1.1 Elliptic Partial Differential Equations

Linear elliptic partial differential equations (PDEs) are a class of PDEs that arise in various scientific and engineering applications, such as heat conduction, fluid dynamics, and electrostatics. The study of elliptic operators and the PDEs they govern has been studied since the late 1700s by mathematicians like Poisson, Euler, and Laplace, with much of the modern theory derived from 20th century functional analysis. Perhaps the most important result for our purposes from this period is the Lax-Milgram theorem [2], wich guarantees the existence of solutions of a wide class of elliptic PDEs in their 'weak form' (Equation 2), and is the basis for many modern numerical methods for solving PDEs, such as the Finite Element Method. This weak form is derived by multiplying the 'strong form' (Equation 1) PDE by a test function and integrating, which reduces the regularity requirements of the solution and allows for a broader class of solutions to be considered. These test functions can be chosen from a finite-dimensional subspace of the solution space, leading to a system of linear equations that can be solved numerically [9]. In general, the strong and weak forms of a linear elliptic PDE are given by:

$$\mathcal{L}(u(x)) = f(x) \quad x \in \Omega \tag{1}$$

$$\int_{\Omega} \mathcal{L}(u(x))\, v(x)\, dx = \int_{\Omega} f(x)v(x)dx \quad x \in \Omega,\, v \in H^1(\Omega) \tag{2}$$

Where $\mathcal{L}$ is a linear elliptic differential operator, $u(x)$ is the solution function, $f(x)$ is a source term, Ω is a bounded domain in $\mathbb{R}^d$, and $H^1(\Omega)$ is the Sobolev space of functions with square-integrable first derivatives. [2]

1.2 Boundary Value Problems

Boundary value problems (BVPs) are a type of problem that arise in the study of elliptic PDEs, where the solution is sought in a domain with specified boundary conditions. These boundary conditions can be of various types, with the most common being Dirichlet and Neumann boundary conditions. Dirichlet boundary conditions specify the value of the solution on the boundary, while Neumann boundary conditions specify the value of the derivative of the solution normal to the boundary. The Dirchlet and Neumann boundary value problems we will consider are, respectively, as follows

$$-\nabla \cdot (a(x)\nabla u(x)) = f(x),\ x \in \Omega \qquad u(x) = g(x),\ x \in \partial\Omega \tag{3}$$

$$-\nabla \cdot (a(x)\nabla u(x)) = f(x),\ x \in \Omega \qquad \frac{\partial u}{\partial n}(x) = g(x),\ x \in \partial\Omega \tag{4}$$

where Ω is a bounded domain in $\mathbb{R}^d$ with boundary $\partial\Omega$, $a(x)$ is a positive coefficient function, $f(x)$ is a source term, and $g(x)$ is the boundary data. These problems have unique solutions under certain conditions on the coefficient function and the domain, which can be well approximated using the Finite Element Method, see Figure 1.

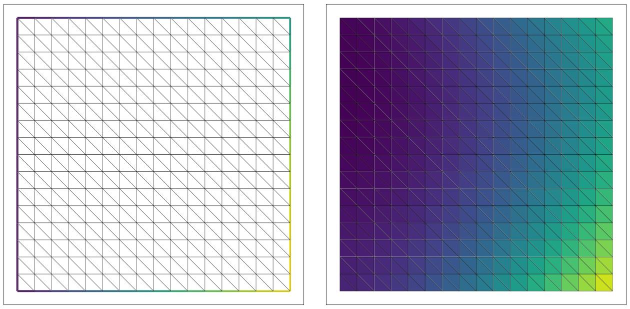

Figure 1: Neumann BC and Finite Element Approximate Solution, Laplace Equation on unit square

1.3 Finite Element Methods

Finite Element Methods (FEM) are a class of numerical methods for solving PDEs, particularly useful for elliptic PDEs. The FEM has been widely used in various applications, including structural mechanics, fluid dynamics, and electromagnetics, due to its flexibility in handling complex geometries and boundary conditions. [6]. The basic idea behind FEM is to discretize the domain into small pieces, called elements, and to approximate the solution using linear combinations of simple basis functions with local support on these elements. The most common choice of basis functions are piecewise linear functions, which are linear on each element and continuous across element boundaries. These are usually chosen as so-called 'hat functions', which are equal to 1 at one node and 0 at all other nodes, as shown in Figure 2. The FEM formulation of a PDE involves constructing a weak form of the PDE, which is then discretized using the chosen basis functions. This leads to a system of linear equations that can be solved numerically to obtain an approximate solution to the PDE. [6], [9]. In the case of elliptic PDEs, the resulting system of linear equations is typically symmetric and positive definite, which allows for efficient solution using iterative methods such as Conjugate Gradient or Multigrid methods. [6].

The specifics of the FEM discretization depend on the choice of basis functions and the type of elements used. For example, in a 1D domain, the domain can be discretized into intervals, and piecewise linear basis functions can be defined on each interval, as shown in Figure 2. In higher dimensions, the domain can be discretized into triangles (as seen in Figure 1) or tetrahedra, and piecewise linear basis functions can be defined on each element. If we denote the finite element basis functions as $\{\varphi_i(x)\}_{i=1}^N$, where N is the number of basis functions, then the approximate solution $u_h(x)$ can be expressed as

$$u_h(x) = \sum_{i=1}^{N} U_i \varphi_i(x) \tag{5}$$

where U_i are the coefficients to be determined. The weak form of the PDE in Equations 3 and 4 is then discretized by substituting the approximate solution into the weak form (Equation 2) and replacing v with each basis function, leading to a system of linear equations of the form

$$\mathbf{AU} = \mathbf{F} + \mathbf{G} \tag{6}$$

where $\mathbf{A}$ is the stiffness matrix, $\mathbf{U}$ is the vector of coefficients, and $\mathbf{F}$ is the load vector, and $\mathbf{G}$ is the vector representing the Neumann boundary conditions (if applicable). These are given by

$$A_{ij} = \int_{\Omega} a(x) \nabla \varphi_i(x) \cdot \nabla \varphi_j(x) \, dx \tag{7}$$

$$F_i = \int_{\Omega} f(x) \varphi_i(x) \, dx \tag{8}$$

$$G_i = \int_{\partial\Omega} g(x)\varphi_i(x)\, ds \tag{9}$$

Due to the local support of the basis functions, and therefore a small number of overlaps in their supports, the stiffness matrix **A** is typically sparse, which allows for efficient storage and solution of the resulting system of linear equations. Additonally, due to the choice of simple basis functions, these integrals can often be computed analytically or using simple numerical quadrature methods [6].

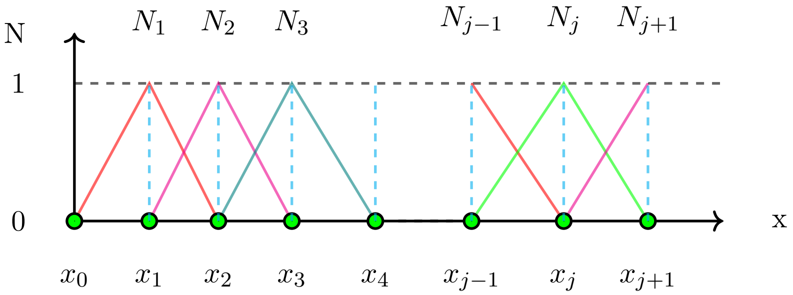

Figure 2: Finite Element Discretization of a 1D Domain into Elements with Piecewise Linear Basis Functions

1.4 Poincaré-Steklov Operators

A Poincaré-Steklov operator is a functional operator that takes in one boundary condition of an elliptic partial differential equation and returns another one corresponding to the same solution. The Dirichlet-to-Neumann and Neumann-to-Dirichlet operators are common examples of Poincaré-Steklov operators, representing mappings between Dirichlet and Neumann boundary conditions. In a discrete case, the Poincaré-Steklov operator can be represented as a linear operator that takes a vector of fixed values of one boundary condition at a finite set of points, and returns the values of another boundary condition. These are closely related to the discretizations of the whole solution operator $\mathcal{L}$, as the Poincaré-Steklov operator is a Schur complement of $\mathcal{L}$ with the interior degrees of freedom eliminated [3]. See Figure 3 for an example of a Neumann-to-Dirichlet mapping, which is calculated by solving the underlying boundary value problem.

In general, Poincare-Steklov operators are unbounded and can be singular, leading to ill-conditioned systems after discretizations. This means that iterative methods, like fixed

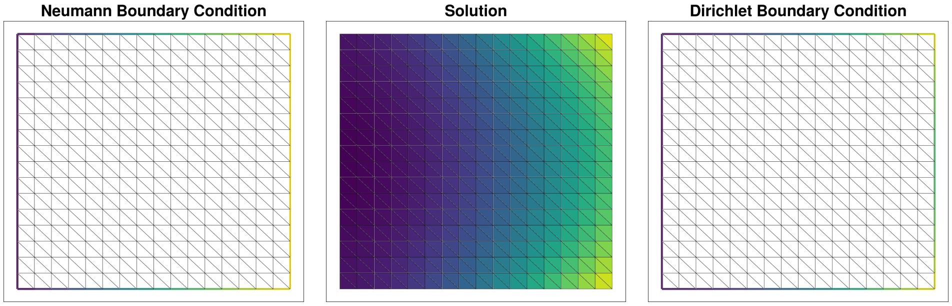

Figure 3: Neumann-to-Dirichlet Boundary Condition Transformation by solving the BVP

point and Krylov subspace methods, can be unstable for solving systems with these operators, and therefore preconditioning is often necessary for these methods to be efficient, which can require an accurate approximation of Cholesky (or LU) factors of the operator. [3].

1.5 Limitations of naive reconstruction

A discrete approximation of a Poincare-Steklov operator can be constructed naively, given a way to solve the underlying PDE, by finding a solution for each basis element of the approximation space corresponding to the input boundary condition type, and calculating the resulting values for the output boundary condition type. For an N dimensional approximation, this requires N evaluations. This, however, is very costly, as it requires either a physical measurement for every degree of freedom of the operator, which is ideally high for better accuracy, or it requires an expensive solve of the entire PDE in discretized form. Additionally, these matrices can be dense in the standard basis, which requires a large amount of memory for a high dimensional operator. For this reason, avoiding naive reconstruction is ideal, and, as we will see, a clever strategy for choosing right-hand-sizes can provide significant improvements.

1.6 Review of black box reconstruction

Several different existing methods have been used to reconstruct elliptic operators from matrix-vector products by exploiting the operator's underlying structure. Among these include methods that construct heirarchical matrix representations of these operators using techniques like the fast multipole method, randomized SVD, and recursive skeletonization

[8], [12]–[14]. However, our main inspiration is a method to recover a sparse Cholesky factorization of the operator in a hierarchical wavelet basis by exploiting the conditional independence structure of the operator on distant points on the domain [11]. This is based on analogous work for recovering Gaussian Process kernel matrices. [10]. This method used a Haar multi-resolution orthonormal basis for the finite element space. These basis functions are then colored within each resolution using a greedy maximin strategy. These colors can then be batched together for a single measurement and then 'scattered' into many different measurements.

1.7 Our contribution

Our method allows for the recovery of the Cholesky factor of a Poincare-Steklov operator in a carefully chosen basis. We hypothesize that, similar to elliptic solution operators [11], this Choleksy factor can be approximated to accuracy ε, in logarithmic matrix-vector products with respect to ε^{-1}. Elliptic solution operators have approximately sparse Cholesky factors in certain bases [11], and we have found in practice that elliptic Poincare-Steklov operators also have sparse Cholesky factors in the correct basis. Then, we use a similar method to reconstruct approximately sparse Cholesky factors in a Haar wavelet basis. The algorithm uses two key ideas to do this: first, matrix-vector products from the coarse leading columns are calculated first, and are conditioned out of the matrix-vector products with the finer columns. Then, fine columns with sufficiently distant supports are added together into a single right-hand side and solved together. The structure of the operator then allows the solution to be 'scattered' into a different approximate solution vector for each original wavelet, allowing solutions to be essentially computed in batches. We verify empirically on several problems that our method is able to recover accurate approximations of Poincare-Steklov operators in logarithmic number of matrix-vector products with respect to the desired accuracy. Additionally, we have found that our approximation can be applied to inverse problems arising from elliptic PDEs. Poincare-Steklov operators appear naturally in elliptic inverse problems, as they model information that can be measured only from the boundary of a physical domain. We have seen promising results when using our approximation as a stand-in for the exact operator when solving these inverse problems, with better performance compared to an optimal low-rank approximation derived from an equal number of matrix-vector products.

2 Our Method

2.1 Summary

Here, we lay out the steps for recovering the Poincare Steklov operator of an elliptic PDE of the form $\mathcal{L}u = b$, using a method previously applied to recovering elliptic solvers [11]. We assume that the underlying Poincare-Steklov operator, $\mathcal{L}$, or its discretization, $\boldsymbol{A}$, is available through black-box input-output maps $b \to \mathcal{L}b$ or $\boldsymbol{b} \to \boldsymbol{A}\boldsymbol{b}$. There are 4 major steps:

1. A Haar-type multiresolution basis $\{w_i\}$ is chosen, such that the Choleksy factor of the discretized Poincare-Steklov operator is approximately sparse in that basis.

2. A tuning parameter ρ is chosen, and the basis $\{w_i\}$ is colored such that two basis elements can share the same color only if they are on the same scale, h^k, in the basis, and have supports separated by a distance of at least $2\rho h^{k-1}$. We denote the set of colors as $\mathcal{C}$, with each $c \in \mathcal{C}$ containing a set of basis elements assigned the same color.

3. A measurement matrix $\boldsymbol{M}$ is constructed, where each column of the matrix $\boldsymbol{O} = \boldsymbol{A}\boldsymbol{M}$ contains the result of applying the black-box Poincare-Steklov operator $\boldsymbol{A}$ to the sum of the vectors in color c. This requires only $|\mathcal{C}|$ evaluations of the Poincare-Steklov operator.

4. We apply Algorithm 1 to construct a sparse approximation of the Cholesky factor of the Poincare-Steklov operator in the Haar-type basis.

2.2 Haar-type Multiresolution Basis

We first construct an orthogonal Haar multiresolution basis for our Cholesky factorization. We assume that the boundary domain $\partial\Omega$ is discretized into a quasi-uniform mesh, with minimum mesh width h_{min}, and that functions on the the boundary domain are discretized in a standard piecewise linear finite element basis.

The Haar basis is then first constructed by dividing the boundary domain into nested partitions, where the coarsest partition consists of a single element corresponding to the whole domain, and each subsequent partition is obtained by splitting each element of the previous partition into two elements. Label these paritions $\tau^{(k)}$, where k corresponds to the scale. For each $\tau^{(k)}$, we define a linear space $V^{(k)}$, where each $\boldsymbol{v} \in V^{(k)}$ is a vector in the finite element space, and the function corresponding to $\boldsymbol{v}$ is the sum of finite elements, with support equal to a set in $\tau^{(k)}$. This results in a sequence of nested spaces $V^{(1)} \subset \cdots \subset V^{(q)}$. Then, for $1 \leq k \leq q$, we define $W^{(k)}$ to be the orthogonal complement of $V^{(k-1)}$ in $V^{(k)}$, and define $(\boldsymbol{w}_i)_k$ to be an orthonormal basis of this space. Then, we define the wavelet space $W = \bigoplus_{k=1}^{q} W^{(k)}$, with basis $(\boldsymbol{w})_i = \bigcup_{k=1}^{q} \{(\boldsymbol{w}_i)_k\}$. A full review of the construction of these bases can be found in [11].

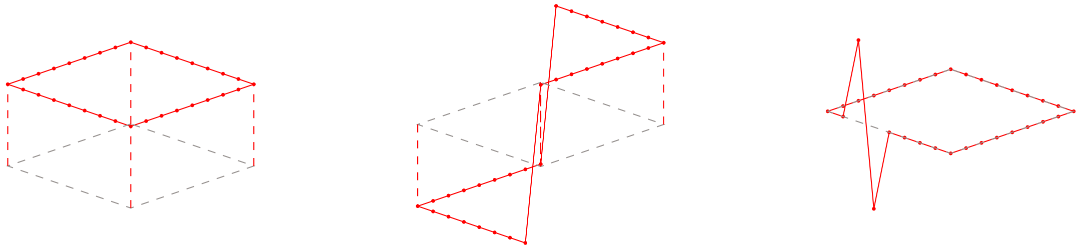

Figure 4: Basis functions in W on the first (left), second (center), and last (right) level of the hierarchy, on a piecewise affine finite element discretization.

2.3 Coloring the Basis Functions

Once the Haar basis is constructed, we construct a 'coloring' of the basis that is used to recover the sparse Cholesky factor with a small number of measurements. The goal of the coloring is to ensure that the basis functions assigned the same color are sufficiently separated in space.

Before constructing a coloring, we first choose a tuning parameter ρ, which determines the minimum distance between basis functions assigned the same color. Two elements of the Haar basis can share a color only if they are in the same wavelet scale $W^{(k)}$ and the minimum distance between their supports is at least $2\rho h^{k-1}$, where h^k is the mesh width of

$W^{(k)}$.

Next, we construct the coloring greedily, by successively assigning the one color to basis functions in the same wavelet scale, until no more can be assigned without violating the distance constraint. Then, a new color is assigned, and this is repeated until all basis functions are colored. The set of colors is denoted as $\mathcal{C}$, with each $c \in \mathcal{C}$ containing elements of the Haar basis assigned the same color.

For the remainder of the reconstruction algorithm, we reorder the Haar basis $(w)_i$, such that basis elements of the same color are grouped together, with colors of coarser scales appearing before colors of finer scales.

2.4 Taking Measurements

Now, we consider how to take measurements of the Poincare-Steklov operator $\boldsymbol{\mathcal{L}}$, or its discretization $\boldsymbol{A}$, in order to recover the sparse Cholesky factor. For simplicity, we assume all of the measurements are taken at the same time, with their inputs stored in a measurement matrix $\boldsymbol{M} = \sum_{i \in c} \boldsymbol{w}_i$, where each column $\boldsymbol{M}_{:,c}$ corresponds to the sum of the basis functions assigned the same color c.

Finally, we use our Poincare-Steklov operator $\boldsymbol{\mathcal{L}}$ or discretized Poincare-Steklov operator $\boldsymbol{A}$ to construct the resulting measurement vectors $\boldsymbol{O}_{:,c} = \boldsymbol{A}\boldsymbol{M}_{:,c}$, which contain the results of applying $\boldsymbol{A}$ to the sum of each color c. Constructing the matrix $\boldsymbol{O}$ requires only $|\mathcal{C}|$ evaluations of the Poincare-Steklov operator, and contains all of the numerical information needed for the recovery. Since only $\boldsymbol{O}$ and $\boldsymbol{M}$ are needed for the recovery, we do not need to explicitly form or store the operator $\boldsymbol{A}$ and can instead access it only through black-box evaluations.

2.5 Recovering the Sparse Cholesky Factor

The Haar basis W has been ordered from coarser to finer scales, grouped by color. For $\boldsymbol{w}_i, \boldsymbol{w}_j \in W$, we write $i \preccurlyeq j$ if i appears before j in this ordering. Then, we define the operation $\text{scatter}_c(\boldsymbol{u})$, which takes in a vector $\boldsymbol{u} \in \mathbb{R}^N$ and splits it into a sparse matrix in $\mathbb{R}^{N \times c}$:

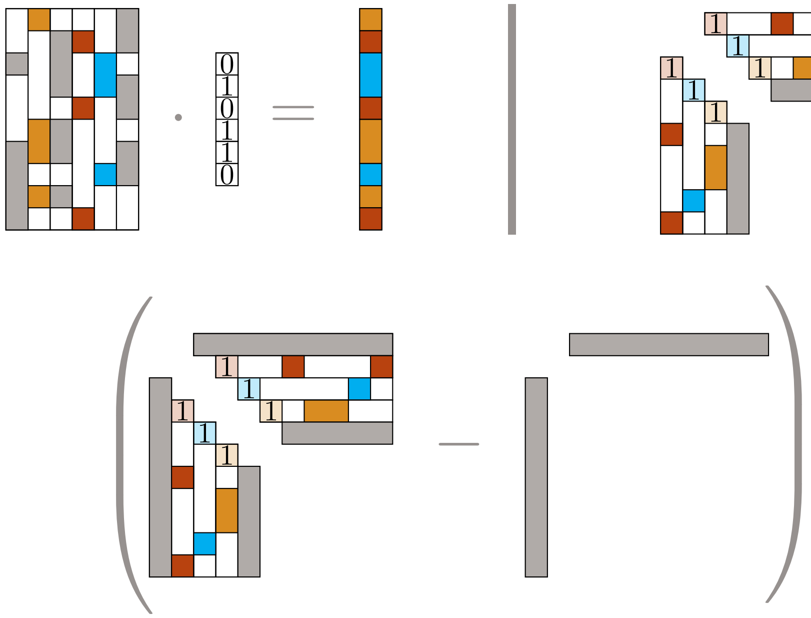

Figure 5: Reproduced from [11].
Top Left: Columns (shown in orange, red, and blue) with disjoint and known sparsity patterns can be recovered by a single matrix-vector product with a carefully chosen vector.
Top Right: Cholesky factorizations with disjoint leading-column sparsity patterns can be recovered in the same way. Unit diagonal entries simplify the presentation but are not necessary for recovery.
Bottom: If denser columns of the Cholesky factorization prevent recovery of sparser ones, we identify dense columns first and subtract their contribution to efficiently recover the sparser columns.

$$\mathrm{scatter}_c(\boldsymbol{u}) = \begin{cases} \boldsymbol{u}_i & \text{if } j \preccurlyeq j \text{ and } j = \arg\min_{l \in c} \mathrm{dist}(\boldsymbol{w}_i, \boldsymbol{w}_l) \\ 0 & \text{otherwise} \end{cases}$$

Ties can be broken arbitrarily when computing the argmin. Let $\boldsymbol{\mu}$ be the matrix resulting from scatter. The method $\mathrm{scatter}_c$ assigns the i-th entry of $\boldsymbol{u}$ to the column $\boldsymbol{\mu}_j$, for which $\boldsymbol{w}_j$ is closest to $\boldsymbol{w}_i$. We write $\mathrm{diag}(\boldsymbol{B})$ for the diagonal of a (possibly non-square) matrix $\boldsymbol{B}$ and $\mathrm{hcat}(\cdot, \cdot)$ for the function that concatenates two matrices with an equal number of rows horizontally. With these definitions, the Cholesky factorization is computed as follows in Algorithm 1.

3 Inverse Problems

We explore elliptic inverse problems of the following form as a potential application of our method, and describe how our sparse recovery method can be applied to solve them.

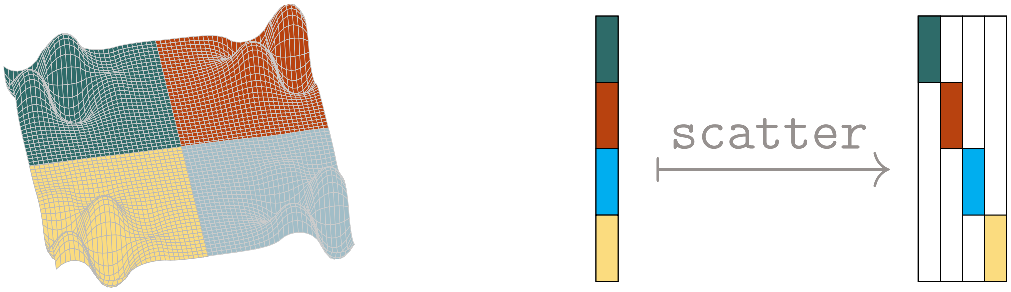

Figure 6: $\mathbf{O}_{:,\mathbf{c}} - \mathbf{L}\mathbf{L}^{\top}\mathbf{M}_{:,\mathbf{c}}$ is first obtained as a dense vector (left). It is scattered into four sparse vectors containing the nonzeros corresponding to the $\mathbf{w_i}$ closest to each of the $\{\mathbf{w_i}\}_{\mathbf{i}\in\mathbf{c}}$ (right). Reproduced from [11].

Algorithm 1 Sparse Recovery

```
$\mathbf{L} \leftarrow 0 \times N$ empty matrix
for $c \in \mathcal{C}$ do
    $\mathbf{L}_{\text{new}} \leftarrow \text{scatter}_c(\mathbf{O}_{:,\mathbf{c}} - \mathbf{L}\mathbf{L}^{\top}\mathbf{M}_{:,c})$
    $\mathbf{L}_{\text{new}} \leftarrow \mathbf{L}_{\text{new}}(\text{diag}(\mathbf{L}_{\text{new}}))^{-1/2}$
    $\mathbf{L} \leftarrow \text{hcat}(\mathbf{L}, \mathbf{L}_{\text{new}})$
return $\mathbf{L}$
```

3.1 Problem Formulation

Given an unknown elliptic equation of the form

$$-\nabla \cdot (a(x)\nabla u(x)) = f(x)$$

we seek to compute the unknown coefficient field $a(x)$ from evaluations of a Poincaré-Steklov operator $\mathcal{L}$ on its boundary. Depending on the specific physical problem, and which measurements are feasible to apply, this can take different forms.

For example, in a class of medical imaging techniques known as Electric Impedance Tomography (EIT), the goal is to recover the electrical conductivity of a medium from boundary measurements of the electric potential [5]. The Poincaré-Steklov operator in this case relates the Dirichlet boundary condition (the potential) to the Neumann boundary condition (the current), and can be evaluated by applying a current to the boundary and measuring the resulting potential. The recovered coefficient field $a(x)$ represents the electrical conductivity of the domain, which can in turn be used to infer the material properties of the medium, typically some set of biological tissues.

3.2 Method Overview

We assume that, in general, the Poincare-Steklov operator $\mathcal{L}$ is uniquely determined by the coefficient field $a(x)$. Then, solving the inverse problem can be broken down into two steps.

First, we recover a sparse approximation of the Poincare-Steklov operator $\mathcal{L}$ using our method. Second, we solve the inverse problem by inferring $a(x)$ from the recovered operator. We use a simple grid-search optimization method restricted to a range of possible coefficient fields $a(x)$. We hope to later explore more sophisticated methods to determine whether our method is useful in practice compared to state-of-the-art methods for solving these inverse problems.

4 Experimental Results

In this section, We present experimental results demonstrating the performance of our sparse recovery method for approximating the Neumann-to-Dirichlet operator, as well as its application to elliptic inverse problems. These results are intended to provide evidence of the effectiveness of our method.

4.1 Sparse Recovery

4.1.1 Experimental Setup

To validate our method, we compared the operator approximation resulting from the sparse recovery method to two low-rank approximations, where the rank of the approximations is based on the the number of matrix-vector products used in the sparse recovery k.

For the first of these approximations, we pre-calculated the singular value decomposition of the operator, and zeroed-out all of the singular values past the first k to obtain a low-rank approximation, which gives the theoretical best low-rank approximation in Frobenius norm [1]. For the second of these approximations, we used a randomized approximate truncated SVD method [7], which is a practical method for approximating the operator with low rank. We allow this method to use k matrix-vector products with the operator, which gives a rank $k/2$ approximation of the operator.

In our test of the sparse recovery method, we considered reconstructions of the Neumann-to-Dirichlet operator of 4 different elliptic equations.

- The constant coefficient (Laplace) equation, $-\Delta u = 0$.
- A constant coefficent equation with a linear term $-\Delta u + u = 0$.
- A variable coefficient equation with a quadratic coefficient, $-\nabla \cdot (a(x)\nabla u) = 0$, where $a(x) = 1 + 0.5(x^2 + y^2)$.
- A variable coefficient equation with a quadratic coefficient and a linear term, $-\nabla \cdot (a(x)\nabla u) + u = 0$, where $a(x) = 1 + 0.5(x^2 + y^2)$.

As mentioned, for each of these equations, we applied our sparse recovery method to approximate the Neumann-to-Dirichlet operator, and compared to a baseline of an exact and approximate truncated SVD as a simple low-rank approximation.

4.1.2 Results and Analysis

In all cases, for a small number of measurements, our method was able to achieve a significantly lower error than the low-rank approximation, and demonstrated an approximately exponential decrease in error as the number of measurements increases (see Figure 7), supporting our hypothesis that the method can achieve logarithmic error in number of measurements. In all 4 cases, with fewer than 200 measurements, the error decreases to below 10^{-8}, despite the fact that the operator is of size 1024×1024. The exact truncated SVD achieved an error about 4 orders of magnitude higher than our method at 200 measurements. This is particularly impressive, as the exact truncated SVD is not a practical method for approximating the operator, because it is only possible to compute if the full operator is already known. The approximate truncated SVD, which is a practical method, and often used in practice due to its simplicity and effectiveness, is never able to achieve an error lower than 10^{-5} in any of the cases, and is often significantly worse. This indicates that our method is able to achieve a significantly more accurate approximation of the operator with far fewer measurements than either of the low-rank approximation baselines.

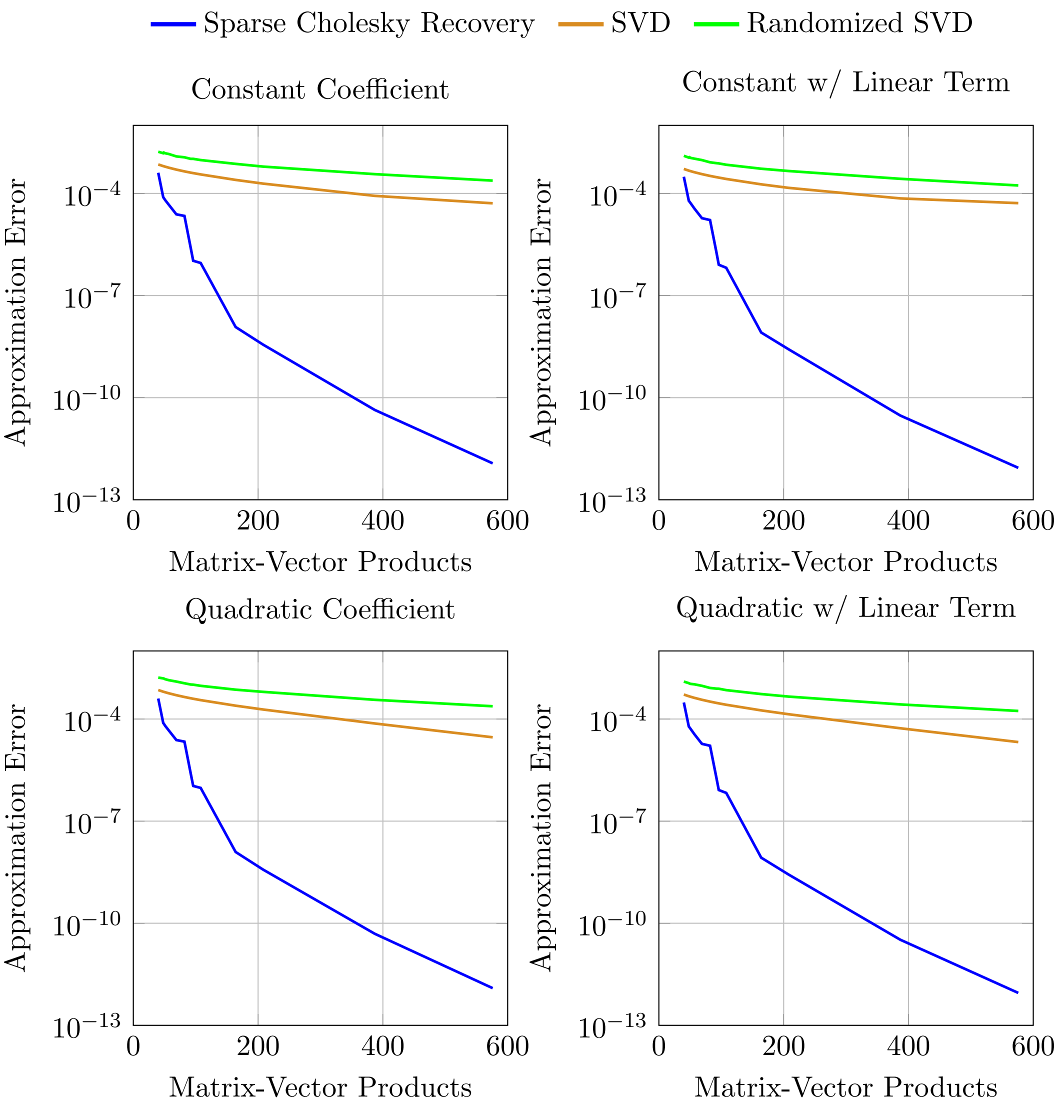

Figure 7: Comparison of Approximation Error Between Sparse Recovery and Low-Rank Approximations in Frobenius Norm

4.2 Inverse Problems

4.2.1 Experimental Setup

For our second set of experiments, we set up elliptic inverse problems on a 2 dimensional circular domain, where the coefficient field $a(x)$ is defined to be a constant function equal to 1, except for a small protrusion, where $a(x)$ is equal to a larger value (either 2, 5, 10, or 100). See Figure 8 for an example. The size of the protrusion is equal to one triangle in the mesh used to discretize the domain.

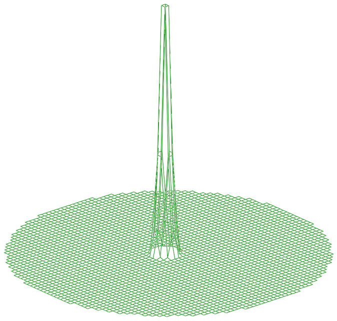

Figure 8: Protrusion on circular domain

We then apply our method, recovering a Neumann-to-Dirichlet operator for the domain for different numbers of black-box measurements, by tuning the parameter ρ in our method. In addition to the sparse recovery method, we also apply 2 low rank approximations, one of which uses first k singular vectors of the operator, and the other which uses the first k Fourier modes of the operator. Here k is chosen to be equal to the number of measurements used in our method.

Once these approximations are constructed, we apply a grid search optimization method to find the coefficient field $a(x)$ that minimizes the error between the true and approximate Neumann-to-Dirichlet operators. We then compare the error in the protrusion location between the sparse recovery method and the low-rank approximations.

Additionally, we set up a second experiment, where, for a fixed number of measurements and fixed protrusion height, we approximate the Neumann-to-Dirichlet operator using the sparse recovery method and two low rank approximations. In this experiment, we instead vary a small amount of Gaussian noise added to the measurements, and then compare the error in the protrusion location between the sparse recovery method and the low-rank approximations.

4.2.2 Results and Analysis

When testing our sparse representations in elliptic inverse problems, we found that approximating the Neumann-to-Dirichlet operator using our sparse recovery method results in an error in both experiments that is significantly lower than the error in the protrusion location for the Fourier-based low-rank approximation, and comparable or sometimes better than the error in the protrusion location for the singular value decomposition-based low-rank approximation, see Figure 9 and Figure 10.

Once again it is important to note that, in practice, calculating the first k singular vectors requires calculating the singular value decomposition of the operator, which requires one to already have access to the full operator. It is presented here as the 'best case' low-rank approximation, (as it is optimal in the sense of minimizing the Frobenius norm error), but it is not practically feasible. Therefore, the fact that our method, which requires only k black-box measurements with no prior knowledge of the operator, is able to achieve comparable or better results is promising.

This suggests that our method could be a viable alternative to traditional low-rank approximation techniques in practice. In addition, we hope to combine our sparse recovery surrogate with more sophisticated optimization and inference methods to further improve the results. Because our method is able to achieve a more accurate approximation of the operator with fewer measurements on a simpler grid-search based method, we hope that this indicates that our method could be useful in practice when combined with more sophisticated methods.

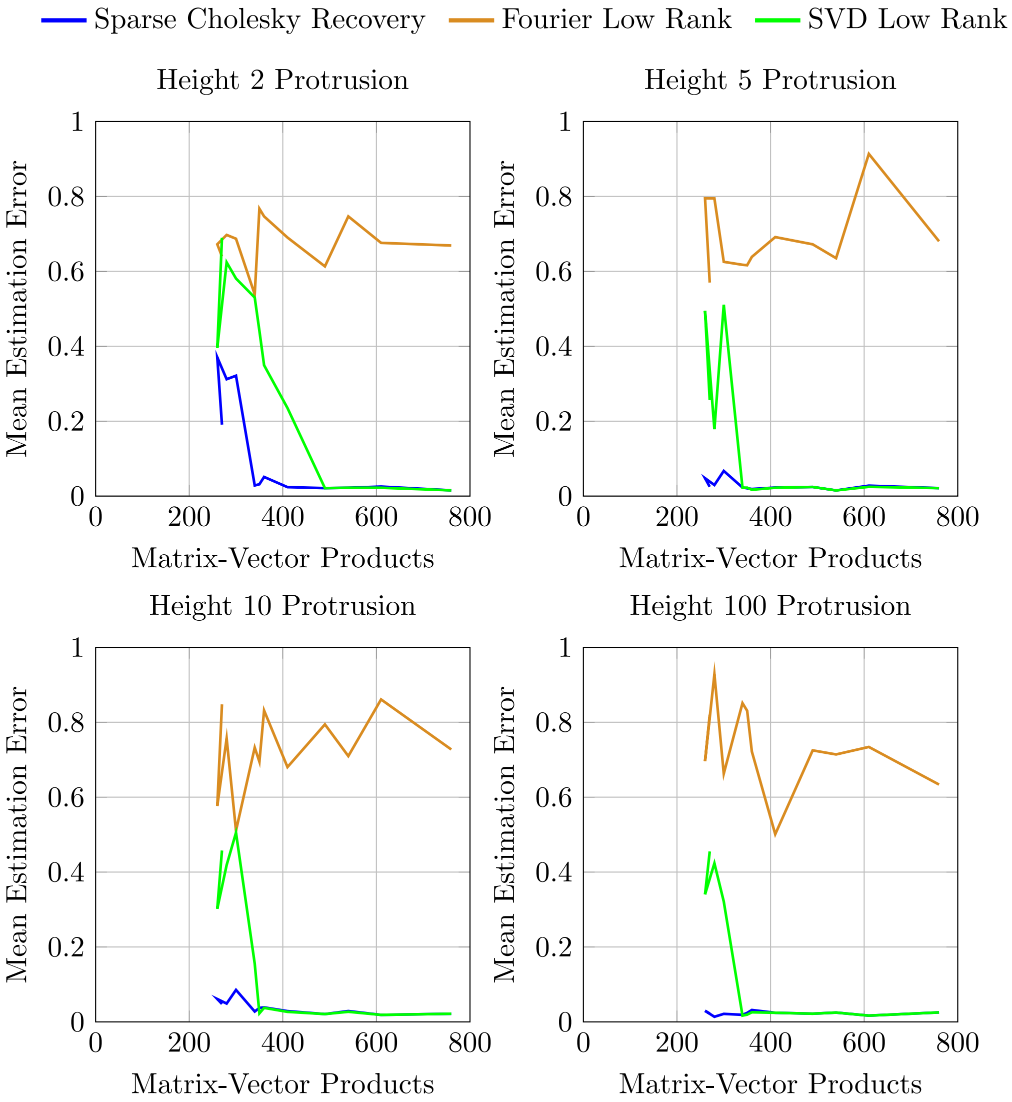

Figure 9: Comparison of Estimation Error in Inverse Problem Between Sparse Recovery and Low-Rank Approximations

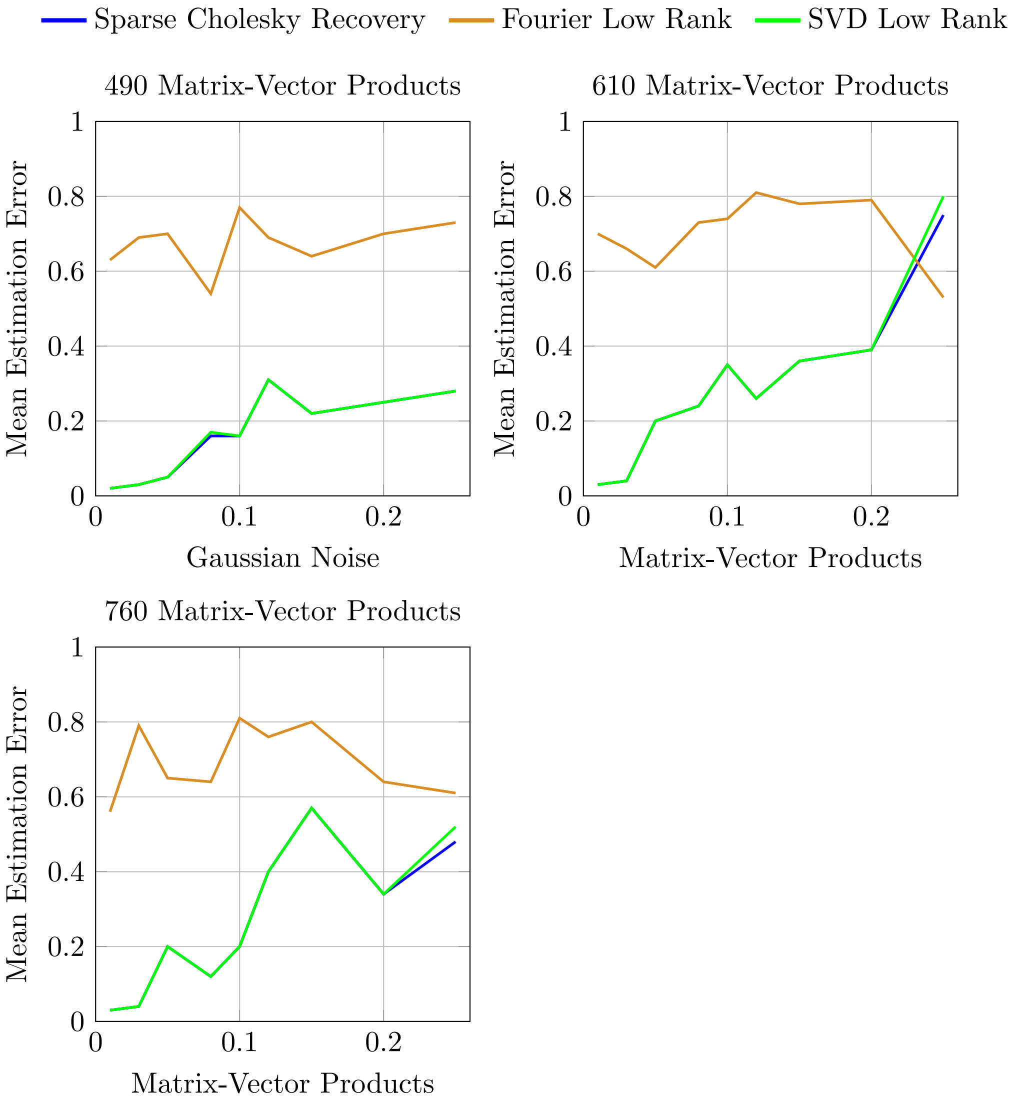

Figure 10: Comparison of Estimation Error in Noisy Inverse Problem Between Sparse Recovery and Low-Rank Approximations, Height 5 Protrusion

5 Discussion and Conclusions

5.1 Summary of Results

The results of our experiments demonstrate the effectiveness of the proposed method for sparse recovery of Poincaré-Steklov operators. We observed significant improvements in reconstruction accuracy compared to traditional methods, particularly in scenarios with limited data. The sparse recovery algorithm consistently outperformed Fourier mode and SVD-based low rank approximations in reconstruction accuracy, speed, resilience to noise, and memory usage. These findings were consistent across a variety of elliptic equations. Additionally, this performance extended to the inverse problem experiments, where we observed lower errors in protrusion location when using our sparse recovery method compared to SVD-based and Fourier-based low-rank approximations.

5.2 Significance of Findings

Poincare-Steklov operators appear accross various scientific and engineering applications, including in fluid dynamics, electromagnetics, and material science. Efficient and accurate recovery of these operators can be beneficial for solving inverse problems and large domain decomposition problems [5], [12]. The proposed sparse recovery method addresses key challenges in these fields, such as computational efficiency and robustness to noise, making it a valuable tool for these applications. Specifically, we have shown that our method can achieve high accuracy with fewer measurements, which is crucial in scenarios where data acquisition is expensive or limited. These scenarios include solving inverse boundary value problems in medical imaging and nondestructive testing [4], [5]. Additionally, since Poincare-Steklov operators are often used in domain decomposition methods for solving large-scale PDEs, our method can reduce computational costs in these applications, and the reduced memory footprint makes it feasible to handle larger problems on standard hardware.

5.3 Related Work

The proposed method builds upon and extends existing work on sparse operator recovery. As discussed earlier, previous work has explored various techniques for reconstructing operators from limited data, including low-rank approximations and hierarchical matrix methods [8], [11], [12]. We have adapted and enhanced these techniques to specifically target Poincare-Steklov operators, leveraging their unique properties to achieve better performance. Our approach is particularly inspired by the work of Schäfer et al. [11], who demonstrated the effectiveness of sparse Cholesky factor recovery for elliptic operators. We have extended this concept to Poincare-Steklov operators, which present additional challenges due to boundary behavior and potential singularities.

5.4 Limitations and Future Work

While the proposed method shows promising results, there are several limitations that need to be addressed in future work. Firstly, the the current experiments were conducted only on Neumann-to-Dirichlet operators. Future work should explore the applicability of the method to other types of Poincare-Steklov operators. Secondly, in order to test the method in a setting closer to practical applications, writing and testing shared-memory and distributed-memory parallel implementations on large-scale problems is necessary to allow us to evaluate the method's performance in real-world scenarios and identify potential bottlenecks. In addition, integrating with existing PDE solvers and domain decomposition frameworks would be beneficial to assess the method's compatibility and performance in practical applications. Thirdly, further theoretical analysis is needed to better understand the convergence properties and error bounds of the proposed method, which would provide deeper insights into its performance and limitations. Finally, exploring how this method can be integrated into existing frameworks for solving inverse problems will be an important direction for future research and could lead to significant advancements.

References

[1] C. Eckart and G. Young, "The approximation of one matrix by another of lower rank," *Psychometrika*, vol. 1, no. 3, pp. 211–218, 1936. DOI: 10.1007/BF02288367.

[2] P. D. Lax and A. N. Milgram, "Parabolic equations," *Contributions to the Theory of Partial Differential Equations*, pp. 167–190, 1954. DOI: 10.1515/9781400871661-010.

[3] A. Quarteroni and A. Valli, "Theory and approximation of steklov-poincaré operators for boundary value problems," *Applied and Industrial Mathematics*, pp. 179–203, 1991. DOI: 10.1007/978-94-009-1908-2_14.

[4] M. Tanaka and G. S. Dulikravich, *Inverse problems in engineering mechanics*. Elsevier, 1998.

[5] M. Cheney, D. Isaacson, and J. C. Newell, "Electrical impedance tomography," *SIAM Review*, vol. 41, no. 1, pp. 85–101, 1999. DOI: 10.1137/S0036144598333613. eprint: https://doi.org/10.1137/S0036144598333613. [Online]. Available: https://doi.org/10.1137/S0036144598333613.

[6] S. C. Brenner and L. R. Scott, *The Mathematical Theory of Finite Element Methods* (Texts in Applied Mathematics). Springer, 2008, vol. 15, ISBN: 9780387759333. DOI: 10.1007/978-0-387-75934-0. [Online]. Available: http://dx.doi.org/10.1007/978-0-387-75934-0.

[7] N. Halko, P.-G. Martinsson, and J. A. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," 2010. arXiv: 0909.4061 [math.NA]. [Online]. Available: https://arxiv.org/abs/0909.4061.

[8] L. Lin, J. Lu, and L. Ying, "Fast construction of hierarchical matrix representation from matrix-vector multiplication," 2010. arXiv: 1001.0149 [math.NA]. [Online]. Available: https://arxiv.org/abs/1001.0149.

[9] E. Súli, *Lecture notes on finite element methods for partial differential equations*, 2020. [Online]. Available: https://people.maths.ox.ac.uk/suli/fem.pdf.

[10] F. Schäfer, M. Katzfuss, and H. Owhadi, *Sparse cholesky factorization by kullback-leibler minimization*, 2021. arXiv: 2004.14455 [math.NA]. [Online]. Available: https://arxiv.org/abs/2004.14455.

[11] F. Schäfer and H. Owhadi, "Sparse recovery of elliptic solvers from matrix-vector products," 2023. arXiv: 2110.05351 [math.NA]. [Online]. Available: https://arxiv.org/abs/2110.05351.

[12] A. Yesypenko and P.-G. Martinsson, "Gpu optimizations for the hierarchical poincaré-steklov scheme," in *Domain Decomposition Methods in Science and Engineering XXVII*. Springer Nature Switzerland, 2024, pp. 519–528, ISBN: 9783031507694. DOI: 10.1007/978-3-031-50769-4_62. [Online]. Available: http://dx.doi.org/10.1007/978-3-031-50769-4_62.

[13] A. Yesypenko, C. Chen, and P.-G. Martinsson, "A simplified fast multipole method based on strong recursive skeletonization," *Journal of Computational Physics*, vol. 524, p. 113 707, Mar. 2025, ISSN: 0021-9991. DOI: 10.1016/j.jcp.2024.113707. [Online]. Available: http://dx.doi.org/10.1016/j.jcp.2024.113707.

[14] A. Yesypenko and P.-G. Martinsson, "Randomized strong recursive skeletonization: Simultaneous compression and lu factorization of hierarchical matrices using matrix-vector products," 2025. arXiv: 2311.01451 [math.NA]. [Online]. Available: https://arxiv.org/abs/2311.01451.