

GPU-Accelerated FGMRES with Overlapping Schwarz Preconditioner for Efficient Solution of Helmholtz Equations

Ziya Mammadov^[0009-0008-2965-6990],
Eero Vainikko^[0000-0003-1902-9383]

1 Introduction

The solution of partial differential equations (PDEs) plays a fundamental role in numerous scientific and engineering applications. Helmholtz equations are of particular significance among the various types of PDEs due to their wide applicability in fields such as acoustics, electromagnetics, quantum mechanics, and seismology, which is the scientific study of earthquakes and elastic waves [1]. However, obtaining accurate and efficient solutions to these equations can be challenging, especially when dealing with large-scale problems or complex geometries. Therefore, iterative Krylov subspace methods [2] have emerged as powerful techniques to solve linear systems arising from discretized PDEs. One of the simplest methods in this class is the Conjugate Gradient (CG) method [3], renowned for its robustness and efficiency for symmetric positive-definite matrices. In the case of unsymmetric matrices, one of the fastest Krylov subspace methods is the GMRES Method (Generalized Minimal Residual Method) [4], which can also be used in the case of more complex non-symmetric preconditioning techniques, like the Domain Decomposition Method (DDM).

1.1 Problem

Solving Helmholtz equations is challenging due to the oscillatory nature of wave propagation and phenomena such as diffraction, interference, and resonance. Accurate solutions require careful treatment of boundary conditions, domain size, and discretization. Moreover, for large wave numbers or complex geometries, the Helmholtz operator becomes severely ill-conditioned, making large-scale problems particularly difficult to solve efficiently while maintaining accuracy and reasonable computational cost.

A variety of alternative approaches have been proposed for the efficient solution of Helmholtz problems. A comprehensive overview of numerical methods for the Helmholtz equation, including multigrid, sweeping, FFT-based, and domain

Ziya Mammadov
University of Tartu, Ülikooli tn 18, 50090 Tartu, Estonia, e-mail: ziya.mammadov@ut.ee

Eero Vainikko
University of Tartu, Ülikooli tn 18, 50090 Tartu, Estonia, e-mail: eero.vainikko@ut.ee

decomposition approaches, is given in [5]. That work highlights the trade-offs between robustness, scalability, and geometric flexibility, particularly in high-frequency regimes, providing context for the Schwarz–Krylov framework adopted in this paper. While geometric and algebraic multigrid methods can be highly effective for low- to moderate-frequency regimes, their performance often deteriorates for high-frequency Helmholtz operators due to the lack of a suitable coarse-grid correction. FFT-based methods and sweeping preconditioners can achieve near-optimal complexity for structured grids and specific boundary conditions, but are less flexible for complex geometries, heterogeneous media, or parallel multi-GPU execution. In contrast, overlapping Schwarz methods combined with flexible Krylov solvers provide a robust, highly parallel framework that naturally accommodates absorption, inexact local solves, and scalable GPU implementations, making them particularly well suited for large-scale high-frequency problems considered in this work. This motivates the use of iterative solvers such as CG and GMRES and the flexible variant of the latter, the Flexible Generalized Minimal Residual method (FGMRES) [3, 4, 9]. However, the indefinite and highly oscillatory nature of the Helmholtz operator leads to poor convergence unless effective preconditioning is employed. Overlapping domain decomposition methods, particularly Schwarz-type preconditioners tailored for high-frequency Helmholtz problems with absorption, have been shown to significantly improve robustness and scalability [6, 7].

1.2 Contribution

In this paper, we study the solution of the high-frequency Helmholtz problem with absorption [6] using the FGMRES method as an outer Krylov subspace method, preconditioned with the Restricted Averaging Additive Schwarz (RAAS) algorithm to enhance FGMRES convergence. The Block Conjugate Gradient [8] is used as an inner method for equal-sized submatrices obtained by applying an RAAS preconditioner to the original domain. This enables the simultaneous solution of multiple subdomains on multiple GPUs, utilizing multiple right-hand side vectors in case the matrices are identical across subdomains.

To enhance the computational speed of these operations, advanced techniques are employed, including the utilization of Graphical Processing Units (GPUs) to accelerate matrix operations using the OpenCL programming framework. Furthermore, numerical experiments will be conducted to evaluate the performance of different solutions in various scenarios using the resources from UT HPC¹. This will involve systematically varying parameters, such as the total number of domains, domain size, and number of inner iterations, to assess the robustness and scalability of the preconditioners. Comparisons will be made based on convergence behavior, computational efficiency, and overall solution accuracy. The outcomes of this investigation will provide insights into the performance gains achieved by leveraging GPU parallel processing and by integrating Domain Decomposition Methods to further enhance the efficiency of the solution process. The objective is to achieve significant speedup and reduce the time-to-solution for large-scale Helmholtz equation problems.

¹ <https://hpc.ut.ee/>

2 Background and Related Work

This section provides an overview of GPGPU programming with OpenCL to accelerate Krylov subspace methods for solving Helmholtz equations, along with background information on the implementation's components.

2.1 Helmholtz Equation and Iterative Solvers

A Helmholtz problem involves finding solutions to the Helmholtz equation, a partial differential equation that models steady-state wave phenomena or vibrations, commonly arising in fields such as acoustics, electromagnetism, and quantum mechanics, where the goal is to determine how physical quantities propagate and resonate spatially within a given domain.

We are solving the equation in a homogeneous medium with uniform wave speed and density throughout the domain, a common assumption in wave propagation models such as acoustics, electromagnetics, and quantum mechanics. In a homogeneous absorbing medium, the material properties are still uniform, but the wave experiences attenuation as it propagates. This is typically modeled by making the wave number a complex number. After discretization, we can solve a finite linear system using iterative methods, such as the FGMRES method [9]:

$$-\Delta u - \left(k^2 + i \cdot \epsilon\right) u = f(x) \quad \text{in } \Omega, \quad (1)$$

- where ϵ is a damping parameter,
- k is a wave number, $k = \frac{\omega}{c}$ where ω is the wave frequency and c is the wave speed.

Discretizing the Helmholtz equation with the finite element method results in a linear system $Au = f$, where A is the so-called Helmholtz matrix.

2.2 Domain Decomposition and Schwarz Methods

The standard Additive Schwarz approach divides the computational domain into overlapping subdomains. In the Helmholtz problem, the simple Additive Schwarz Method with minimal overlap performs poorly. Increasing overlap between subdomains can improve the convergence rate of iterative solvers by enabling smoother information transfer and reducing interface errors. Each subdomain solves its local problem on the extended domain.

The Restricted Additive Schwarz (RAS) method introduces a restriction operator [7], limiting updates after the subdomain solution to internal nodes only. The Additive Average Schwarz (AAS) method updates overlap values by averaging contributions from neighboring subdomains, rather than summing them up. The Restricted Additive Average Schwarz (RAAS) method averages only within the minimal overlap, while using exterior values exclusively as input for the extended subdomain problem.

2.3 GPU and Multi-GPU Solvers

While modern GPU libraries (cuSPARSE, MAGMA, AmgX, PETSc) provide efficient SpMV and Krylov subspace routines, they often lack support for custom preconditioners, flexible management of subdomain overlap, or efficient simultaneous solves for multiple right-hand sides - key features in advanced domain decomposition techniques.

To address these limitations, our work uses custom iterative solvers implemented with PyOpenCL² — a Python library that enables OpenCL-based programming for heterogeneous systems, enabling the creation of high-performance, portable code that runs on various hardware platforms. It manages kernel launches, memory operations, and data exchanges, providing fine-grained control needed for subdomain solves, overlap updates, and asynchronous multi-GPU communication. This enables simultaneous processing of multiple right-hand sides with identical local matrices, thus maximizing GPU efficiency and scalability.

3 Methodology

We employ the RAAS method to solve the problem, which is divided into equal-sized and equally shaped subdomains with extended overlaps. This approach employs a tile-based domain decomposition method [11] with identical overlapping tiles that cover the entire 2D domain. In the case of a homogeneous Helmholtz problem, this results in a set of subdomain problems with identical sparse matrices. We solve these using Block Conjugate Gradient, where we utilize the multiple GPUs to solve the multiple RHSs simultaneously, see Fig. 1

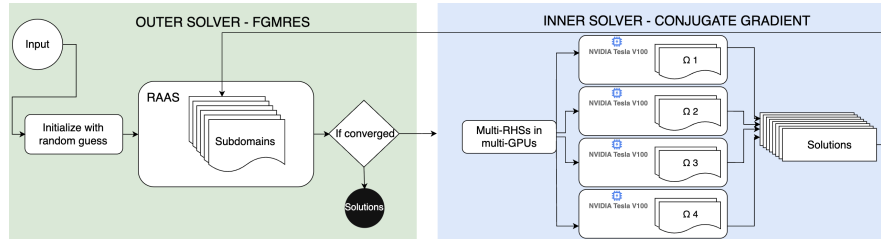


Fig. 1 Helmholtz Solver

3.1 Outer solver - FGMRES preconditioned with RAAS

At the outer iteration level, FGMRES facilitates the use of a variable preconditioner at each iteration. A slightly different preconditioner may be obtained at every FGMRES step because each subdomain is solved inexactly using CG, and the number of CG steps may vary depending on convergence tolerances. The RAAS preconditioner divides the domain into overlapping subdomains with overlap δ ; each subdomain is solved and mapped to a GPU, with local boundary exchanges managed via MPI.

² <https://pypi.org/project/pyopencl/>

Additional variation arises from using different GPUs for different subdomains, where hardware-specific rounding may introduce minor discrepancies.

For homogeneous media, our RAAS-based domain decomposition preconditioner produces subdomains with similar boundary conditions.

3.2 Inner Solver - GPU-Accelerated Block Conjugate method

Each subdomain is solved independently using a Conjugate Gradient method implemented via custom OpenCL kernels on multiple GPUs. This approach leverages GPU parallelism for core computation steps, including sparse matrix-vector multiplication, vector updates, inner products, and tolerance checks, thereby enabling an efficient, scalable solution to the decomposed Helmholtz systems and rapid updates to subdomain solutions for the outer Krylov solver.

4 Experimental Setup

Numerical experiments were conducted on a high-performance computing cluster equipped with multiple NVIDIA Tesla V100-PCIE-32GB GPUs. Each GPU was used to solve subdomain problems in parallel, with the PyOpenCL framework facilitating efficient kernel management and host-device communication. All experiments ran on Ubuntu 22.04 with Python 3.11, using OpenCL drivers provided by NVIDIA.

Benchmarks included two-dimensional Helmholtz test problems, discretized using second-order finite elements and subject to impedance boundary conditions. The computational domain was divided into up to 100 subdomains, with subdomain sizes ranging from 32×32 to 64×64 . The damping number ϵ and wavenumber k were set to 20. Overlap sizes were systematically varied from 0 up to half the subdomain width. The number of inner CG iterations and wavenumber values were also adjusted across experiments to study convergence behavior under different physical and computational regimes.

5 Results

We evaluated FGMRES-based domain decomposition solvers on single- and multi-GPU architectures. Performance metrics are reported for various solver implementations and domain configurations, highlighting the impact of parallelism.

5.1 Single-GPU Performance

We have compared the performance of SciPy Exact subsolves, PyOpenCL single-RHS solver, and PyOpenCL multiple-RHS solver in various setups, as shown in Table 1.

PyOpenCL-CG with multiple-RHS consistently outperforms the other solvers, having the lowest total runtime in every setup. For example, in the simplest case (16 subdomains, 64^2 size), it completes in 32 seconds, compared to 77 seconds for SciPy and 81 seconds for PyOpenCL with a single RHS. As problem size scales up (largest matrix 1,306,449), all solvers slow down substantially, but the relative advantage of PyOpenCL-CG multiple RHSs remains significant (1849s runtime vs. 4572s for SciPy)

Table 1 Solver performance for different subdomain setups. Time is total runtime (s), Iter. is the number of outer FGMRES iterations, Avg. is the average time per FGMRES iteration (s).

Setup			SciPy			PyCL single RHS			PyCL multi-RHS		
#Subs	Size	Mtr	Time	Iter.	Avg	Time	Iter.	Avg	Time	Iter.	Avg
16	64 ²	99,225	77	19	3.6	81	22	3.2	32	22	1.0
16	128 ²	403,225	543	19	26.5	312	27	10.1	229	27	7.0
16	256 ²	4,162,625	4667	20	225.7	1810	28	58.9	1703	28	54.8
25	32 ²	34,596	22	24	0.7	74	26	2.8	9	26	0.2
25	64 ²	142,884	153	25	5.5	132	30	3.9	60	30	1.46
25	128 ²	580,644	917	25	33.9	616	31	17.8	436	31	12.0
36	32 ²	47,089	29	29	0.8	142	33	4.1	18	33	0.4
36	64 ²	194,481	226	30	6.8	312	35	8.2	105	35	2.3
36	128 ²	790,321	1631	31	49.5	1013	38	24.4	684	38	15.8
64	32 ²	77,841	75	40	1.61	314	46	6.64	38	46	0.6
64	64 ²	321,489	615	41	14.0	575	47	11.4	209	47	3.6
64	128 ²	1,306,449	4572	42	105.2	2500	59	39.7	1849	59	28.6

5.2 Multi-GPU Performance

Experiments were conducted with varying subdomain numbers and sizes. Subdomain overlap Δ was set to maximum. 4 NVIDIA Tesla V100 GPUs attached to the same cluster node were employed to run the experiments using multiple RHSs.

Table 2 Multi-GPU solver performance on different subdomain setups for Helmholtz solution

# Subdomains	Subdomain Size	Inner Iterations	Time (s)
100	64 × 64	512	393
100	32 × 32	128	55
64	32 × 32	128	28

The number of inner iterations was set to 4 times the subdomain size, which is sufficient for the solver to converge to the solution. Similar tests were conducted, including residual checks for the inner solver, which were performed on GPUs; however, the solver's performance became slightly slower due to the additional computations, especially for smaller subdomain sizes. Therefore, we chose the GPU solver version without residual checks for this experiment.

Table 2 demonstrates that the multi-GPU solver successfully overcomes single-GPU memory limitations, enabling the solution of larger subdomain setups for the Helmholtz problem.

5.3 Influence of the wave number on the convergence

For the experiment, we have set the total number of subdomains to 16 and the size of each subdomain to 128. Moreover, the subdomain overlap size was set to 31, and the damping number ϵ to 20. The wavenumber varied from 5 to 150.

The execution time increases with wavenumber from 5 to 50, as expected due to the problem's increasing complexity. Higher wavenumbers in Helmholtz equations

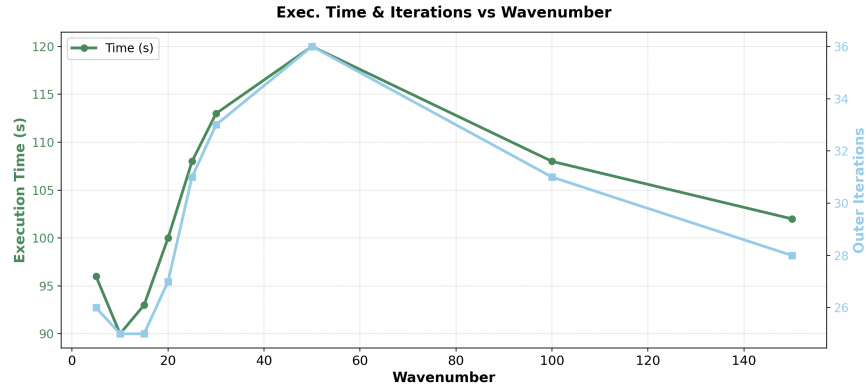


Fig. 2 Wavenumber vs. Outer iterations and Execution Time

often make the problems more oscillatory and harder to solve due to their impact on the eigenvalue distribution of the matrix. However, it is surprising that the solver solves the problem faster as the wavenumber increases to 100 and 150. This behavior of the solver will be investigated further.

5.4 The impact of subdomain overlaps on the performance of solvers

The experiments used 16 subdomains of size 64, 256 CG iterations, wavenumber $k = 20$, and damping parameter $\varepsilon = 20$.

The time-to-solution initially decreases as the number of subdomain overlap sizes increases from 1 to 13, indicating improved solver efficiency with moderate overlaps. As the overlap width increases, the number of outer iterations decreases, while the size of the subdomain problem grows. Beyond 13 layers, the execution time increased gradually, reflecting the diminishing returns in efficiency at higher overlaps.

Overall, the data suggests that increasing the subdomain overlap can initially reduce computational complexity. Especially up to the point where the overlap doesn't overlap with the third subdomain overlap. In order to avoid such situations, we have a formula $(subdomain_width/2 - 1)$ to calculate the maximum subdomain overlap without extending over the second subsequent subdomain.

6 Future Work

As future work, we plan to conduct a more thorough investigation of the implemented algorithms on a diverse set of hardware configurations to optimize computational and parallel scalability on these platforms. The possible parameters for implementing the RAAS method depend on the type and number of GPUs to be used, as well as the available memory size on these devices. In particular, we aim to optimize the method's performance with respect to the number of subdomains to use on each GPU, as well as the number of CG iterations during subsolves on each subdomain. Furthermore, we will investigate how to adapt the solvers to handle uneven subdomains by introducing a restriction and an interpolation operator. Moreover, applying

AI techniques to dynamically select a preconditioner based on real-time analysis of the solver's performance can accelerate the solver.

Acknowledgements This work was supported by the Institute of Computer Science and the HPC center of the University of Tartu, as well as related research grants.

References

1. Juraev, D.A., Agarwal, P., Elsayed, E.E., Targyn, N.: Applications of the Helmholtz equation. *Adv. Eng. Days (AED)* **8**, 28–30 (2023)
2. Liesen, J., Strakoš, Z.: *Krylov subspace methods: principles and analysis*. Oxford University Press, 2013
3. Shewchuk, J.R.: An introduction to the conjugate gradient method without the agonizing pain. Technical Report, Carnegie Mellon Univ. (1994)
4. Walker, H.F., Zhou, L.: A simpler GMRES. *Numer. Linear Algebra Appl.* **1**(6), 571–581 (1994)
5. Gander, M.J., Zhang, H.: A class of iterative solvers for the Helmholtz equation: factorizations, sweeping preconditioners, source transfer, single layer potentials, polarized traces, and optimized Schwarz methods. *SIAM Rev.* **61**(1), 3–76 (2019)
6. Graham, I., Spence, E., Vainikko, E.: Domain decomposition preconditioning for high-frequency Helmholtz problems with absorption. *Math. Comput.* **86**(307), 2089–2127 (2017)
7. Efsthathiou, E., Gander, M.J.: Why restricted additive Schwarz converges faster than additive Schwarz. *BIT Numer. Math.* **43**(5), 945–959 (2003)
8. Tali, K.: Block conjugate gradient solver in OpenCL. University of Tartu, DSSem 2021 Spring (2021). Available via course page. <https://courses.cs.ut.ee/2021/dssem/spring/Main/HomePage?action=download&upname=tali-cg-opencl.pdf>
9. Saad, Y.: A flexible inner-outer preconditioned GMRES algorithm. *SIAM J. Sci. Comput.* **14**(2), 461–469 (1993)
10. Raju, M.P., Khaitan, S.: Domain decomposition based high performance parallel computing. *Int. J. Comput. Sci. Issues* **5**, 27–32 (2009)
11. Gropp, W.D., Keyes, D.E.: Domain decomposition with local mesh refinement. *SIAM J. Sci. Stat. Comput.* **14**(4), 967–993 (1992)