

Acceleration of Algebraic Optimized Schwarz Methods Using a Single GPU Computations

Safae Bourhnane^[0000-0002-1785-5326], Lahcen Laayouni^[0009-0000-5378-2825], and Imad Kissami^[0000-0001-6740-1085]

1 Introduction

Solving large-scale linear systems is a key challenge in scientific computing, particularly in numerical simulations of partial differential equations (PDEs). These systems, which arise from discretizing PDEs using methods like finite differences, finite elements, or finite volumes, are often large, sparse, and ill-conditioned [1, 2].

Efficient solvers are crucial for handling large-scale linear systems. Direct solvers are impractical for large, sparse systems due to their high computational cost and memory requirements [3]. This paper shows that iterative solvers are more effective for these systems, as they exploit sparsity and reduce computational demands. Domain decomposition methods further enhance this efficiency by dividing problems into smaller, parallelizable subproblems.

The Algebraic Optimized Schwarz Methods, as proposed in [4], are based on the concept of replacing the transmission blocks A_{22} and A_{33} with modified blocks: $S_1 = A_{33} + D_1$ and $S_2 = A_{22} + D_2$. This idea draws inspiration from the Optimized Schwarz Methods (OSM) [5]. The approach of identifying the optimal blocks D_1 and D_2 mirrors the process of determining the best parameters in Optimized Schwarz Methods. Several papers have demonstrated that these variants significantly improve the convergence rates of such domain decomposition methods [4], and related techniques have been successfully used in a wide range of numerical applications, such as in Cai and Sarkis (1999) [6].

Safae Bourhnane

Adolf-Reichwein-Straße 23 07745, Jena, Germany. e-mail: safae.bourhnane@leibniz-hki.de

Lahcen Laayouni

School of Science and Engineering, Al Akhawayn University, Avenue Hassan II, 53000 P.O. Box 1630, Ifrane, Morocco e-mail: L.Laayouni@au.ma

Imad Kissami

College of Computing, Mohammed VI Polytechnic University, Lot 660, Benguerir, Morocco e-mail: imad.kissami@um6p.ma

Several studies have also explored the implementation of Schwarz-type methods on GPUs. Liu et al. developed a GPU-accelerated Restricted Additive Schwarz (RAS) preconditioner using ILU factorization and a parallel triangular solver [7]. However, their work highlights challenges related to the inherently sequential nature of triangular solves, which limits parallel efficiency on GPUs and introduces memory access irregularities. Osaki and Ishikawa implemented a GPU-based domain decomposition solver in the context of lattice QCD simulations, using mixed-precision BiCGStab solvers with RAS preconditioners [8]. They identified significant communication overhead and load imbalance between subdomains as primary bottlenecks, particularly when scaling across multiple GPUs. Nayak et al. introduced an asynchronous Schwarz algorithm using MPI RMA communication and evaluated its performance on heterogeneous GPU-CPU platforms [9]. While their asynchronous approach reduced global synchronization, it also introduced complexity in ensuring numerical stability and convergence, especially under high-latency interconnects.

Our study focuses on leveraging GPUs to accelerate Algebraic Optimized Schwarz Methods (AOSM) when used as preconditioner solvers on GPU devices. We compare the performance of the optimal algebraic algorithm with that of the classical Schwarz method. The study includes examples involving various differential equations and discretization methods. The tolerance for the optimal algorithm to converge in two iterations is set at 10^{-6} , and we consider the additive version of the algorithm.

Unlike prior GPU-accelerated Schwarz implementations that focus on classical RAS preconditioners or asynchronous execution strategies, our approach targets algebraic optimization of the transmission conditions. Specifically, we compute the optimal transmission blocks D_1 and D_2 using GPU-accelerated sparse-dense operations, in combination with MPI-enabled sparse solvers. This allows us to construct transmission matrices that yield convergence in two iterations under ideal conditions. In contrast to methods relying solely on GPU-accelerated ILU or asynchronous domain solves, our method achieves high-accuracy convergence while selectively utilizing GPU resources for the most computationally intensive components.

Numerical experiments are conducted on several representative model problems, including Laplacian and Advection-Reaction-Diffusion equations, and are carried out on modern GPU architectures, namely the NVIDIA A100. The results demonstrate substantial performance gains over CPU-based implementations, particularly for large-scale problems, highlighting the effectiveness of combining AOSM with GPU acceleration.

The paper is organized as follows: Section 2 provides a brief description of AOSM Methods. Section 3 presents the main contributions of the paper, demonstrating the acceleration of AOSM with GPUs for various models. Section 4 concludes with remarks on the challenges and findings of this work.

2 Algebraic Optimized Schwarz Methods

The goal of Algebraic Optimized Schwarz Methods (AOSM) is to address the problem of solving linear systems characterized by the equation

$$Au = f, \quad (1)$$

where A is an $n \times n$ matrix with a block-banded structure:

$$A = \begin{pmatrix} A_{11} & A_{12} & & & \\ A_{21} & A_{22} & A_{23} & & \\ & A_{32} & A_{33} & A_{34} & \\ & & & A_{43} & A_{44} \end{pmatrix}, \quad (2)$$

where the submatrices A_{ij} have dimensions $n_i \times n_j$, with $i, j = 1, \dots, 4$, and $n = \sum_i n_i$. In the context of discretized partial differential equations (PDEs), we assume that $n_1 \gg n_2$ and $n_4 \gg n_3$, indicating that certain blocks of the matrix are significantly larger than others. In typical PDE scenarios, the large blocks A_{11} and A_{44} represent the degrees of freedom in the interior of subdomains, while the smaller blocks A_{22} and A_{33} correspond to interface (and possibly overlap) unknowns.

To understand the asymptotic behavior of AOSM accelerated by GPUs, we focus on Optimized Restricted Additive Schwarz (ORAS). We also provide Optimized Restricted Multiplicative Schwarz (ORMS) for completeness. The error propagation operators for these methods are given by (see [4]):

$$T_{\text{ORAS}} = I - \sum_{i=1}^2 \tilde{R}_i^T \tilde{A}_i^{-1} R_i A, \quad T_{\text{ORMS}} = \prod_{i=2}^1 \left(I - \tilde{R}_i^T \tilde{A}_i^{-1} R_i A \right), \quad (3)$$

where R_i and \tilde{R}_i are the classical restriction matrices used in Restricted Additive Schwarz (RAS) methods [6], with and without overlap, respectively. The matrices \tilde{A}_1 and \tilde{A}_2 represent modified subdomain matrices used in these optimized methods [10] and are defined as follows:

$$\tilde{A}_1 = \begin{pmatrix} A_{11} & A_{12} & & \\ A_{21} & A_{22} & A_{23} & \\ & A_{32} & S_1 & \end{pmatrix}, \quad \tilde{A}_2 = \begin{pmatrix} S_2 & A_{23} & & \\ A_{32} & A_{33} & A_{34} & \\ & & & A_{43} & A_{44} \end{pmatrix}, \quad (4)$$

with $S_1 = A_{33} + D_1$ and $S_2 = A_{22} + D_2$. Here, D_1 and D_2 are transmission matrices chosen to improve the convergence rate.

As shown in [4], the asymptotic convergence factor of AOSM depends on the product of the norms:

$$\|(I + D_1 B_{33})^{-1} (D_1 B_{12} - A_{34} B_{13})\|, \quad \text{and} \quad \|(I + D_2 B_{11})^{-1} (D_2 B_{32} - A_{21} B_{31})\|, \quad (5)$$

where the blocks B_{ij} are derived from the columns of A_1^{-1} and A_2^{-1} , for more details, see [4]. Optimal choices for D_1 and D_2 can be derived to minimize the norms in (5),

leading to:

$$D_{1,\text{Opt.}} = -A_{34}A_{44}^{-1}A_{43}, \quad \text{and} \quad D_{2,\text{Opt.}} = -A_{21}A_{11}^{-1}A_{12}. \quad (6)$$

These choices ensure that the AOSM method converges in just two iterations under ideal conditions with two subdomains. However, computing $D_{j,\text{Opt.}}$ requires solving linear systems involving A_{44} and A_{11} , which is computationally intensive. In our implementation, we solve these systems in parallel using the MUMPS direct solver [11], which internally relies on MPI. The resulting sparse-dense matrix products needed to compute the transmission blocks are then accelerated on a single GPU. Thus, the parallelism is handled by MUMPS for the subdomain solves, while the GPU is used to accelerate the most computationally expensive algebraic operations.

3 Numerical results

For all the numerical experiments provided in this section, we used NVIDIA A100 on Toubkal Supercomputer [12], and the AOSM algorithm is used as a preconditioner for the GMRES method without restart.

3.1 Laplacian model problem

Our first model problem is the Laplacian equation in two-dimensional space. The domain is $\Omega = (-1, 1)$. Using a finite difference method, the resulting matrix A is of size 1024×1024 . It is decomposed into two subdomains, with blocks A_{11} , A_{12} , A_{21} , and A_{22} of sizes 480×480 , 480×32 , 32×480 , and 32×32 , respectively.

Table 1: Comparison of CPU and GPU Execution Times (in seconds) for Laplacian model problem.

Matrix Size	CPU timing		GPU timing	
	Schwarz	Additive	Schwarz	Additive
1,000	0.1334	0.4136	0.0900	0.1428
4,000	3.3318	6.1048	0.9138	0.9935
16,000	122.8066	157.6440	15.1334	15.4259
32,000	807.8815	942.3812	53.4316	54.2041
64,000	> 13h	> 13h	515.2441	443.3986

As shown in Table 1 and Figure 1, GPU acceleration significantly reduces execution time for the Laplacian model. For a problem size of 1,000, we observe speedups of approximately $1.48\times$ for the Schwarz method and $2.90\times$ for the Additive method. At 64,000, these speedups increase dramatically to about $90.83\times$ and $105.55\times$, re-

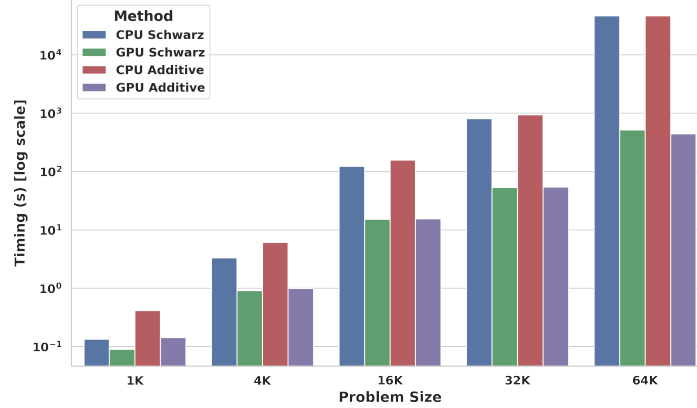


Fig. 1: Speedup for the Laplacian problem (log scale).

spectively. Both methods benefit from GPU use, though the Additive method incurs slightly higher computational cost due to additional operations.

3.2 Advection-Reaction-Diffusion Equation

Our second model we consider the advection-reaction-diffusion equation:

$$\eta u - \nabla \cdot (a \nabla u) + b \cdot \nabla u = f, \quad (7)$$

where $a = a(x, y) > 0$, $b = [b_1(x, y), b_2(x, y)]^T$, and $\eta = \eta(x, y) \geq 0$, with

$$b_1 = y - \frac{1}{2}, \quad b_2 = -x + \frac{1}{2}, \quad \eta = x^2 \cos^2(x + y), \quad a = 1 + (x + y)^2 e^{x-y}.$$

The right-hand side function is $f = 0$ and we use a standard five-point centered finite difference discretization, which results in a matrix A of size 1024×1024 . This matrix is decomposed according to the subdomain decomposition specified in (2), with $n_1 = 480$, $n_2 = 32$, $n_3 = 32$, and $n_4 = 480$.

Similar to Laplacian model problem, and as seen in Table 3 and Figure 2, GPU acceleration leads to considerable performance improvements for the Advection-Reaction-Diffusion model. At a problem size of 1,000, we observe speedups of approximately $1.87\times$ for the Schwarz method and $4.00\times$ for the Additive method. These speedups increase significantly for the largest problem size of 64,000, reaching about $16.75\times$ and $20.35\times$, respectively. This demonstrates the growing advantage of GPU acceleration as problem size increases.

Table 2: CPU vs GPU Execution Times (in seconds) for Advection-Reaction-Diffusion problem.

Matrix Size	CPU timing		GPU timing	
	Schwarz	Additive	Schwarz	Additive
1,000	0.0981	0.3797	0.0526	0.0949
4,000	3.3172	6.0601	1.1011	1.1631
16,000	132.2776	168.9234	20.3437	19.8778
32,000	795.5596	930.0826	69.1177	70.5975
64,000	7367.1952	8153.2579	439.7854	400.6419

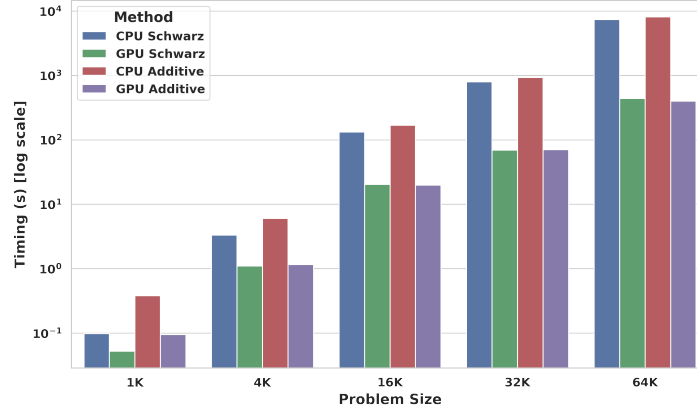


Fig. 2: Speedup for Advection-Reaction-Diffusion problem log scale).

3.3 Finite Volume Matrix

The finite volume matrix arises from the discretization of the Laplace equation in a domain $\Omega \subset \mathbb{R}^2$ using the Diamond Scheme [13].

Let $u(x, y)$ denote the temperature at $(x, y) \in \Omega$. u is the solution of the problem:

$$\begin{cases} \Delta u(x, y) = 0, & \text{for } (x, y) \in \Omega, \\ u(x, y) = u_b, & \text{for } (x, y) \in \partial\Omega. \end{cases} \quad (8)$$

Table 3 and Figure 3 depict the CPU time (s) and the speedup of the additive and the Schwarz and Additive for solving Laplace problem using a FVM discretization using Diamond scheme. The GPU acceleration significantly reduces execution time for the Laplace problem discretized by the finite volume method using the Diamond scheme. At 1,000 unknowns, we observe speedups of approximately $2.23\times$ (Schwarz) and $4.38\times$ (Additive). For the largest case with 64,000 unknowns, the speedups reach $14.62\times$ and $17.64\times$, respectively.

Table 3: CPU vs GPU Execution Times (in seconds) for Laplace problem using FVM with Diamond Scheme.

Matrix Size	CPU timing		GPU timing	
	Schwarz	Additive	Schwarz	Additive
1,000	0.07825	0.3820	0.03510	0.08714
4,000	3.6986	6.9797	0.8792	0.9817
16,000	122.6774	174.0718	12.4442	12.7958
32,000	907.5456	1148.9173	114.8017	115.8191
64,000	6865.9390	8191.9172	469.5820	464.1481

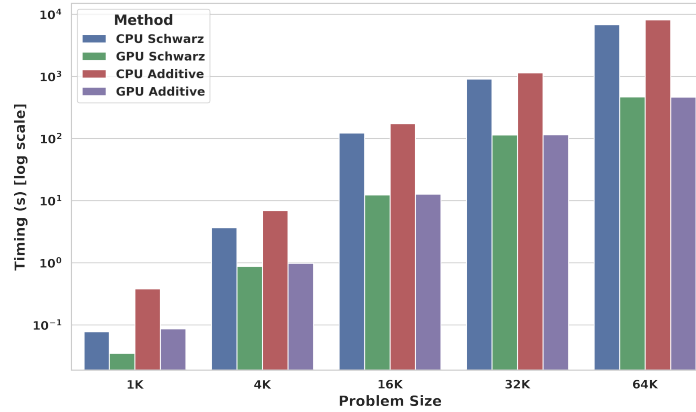


Fig. 3: Speedup for Laplace problem using FVM with Diamond Scheme.

4 Concluding remarks

The additive method consistently outperforms the Schwarz method for both CPU and GPU implementations. GPU acceleration provides substantial speedup, particularly for larger matrices. Execution times vary across different problem domains, so the choice of method and scheme should align with the specific characteristics of the problem. Overall, GPU acceleration delivers a significant speedup compared to CPU execution.

This paper considers only the optimal Schwarz algorithm, which becomes very expensive for large-scale problems. For more practical situations and industrial problems, one should consider techniques developed in [14, 15]. These promising algorithms use SPAI approximation to estimate the blocks D_1 and D_2 .

In future work, we plan to further enhance the performance of Algebraic Optimized Schwarz Methods (AOSM) on GPUs by integrating cuBLAS and other GPU-accelerated libraries. We will also explore advanced techniques like mixed-precision arithmetic and parallel algorithm tuning to improve efficiency and scalability for larger, more complex problems.

References

1. K. W. Morton, D. F. Mayers, Numerical solution of partial differential equations: an introduction, Cambridge University Press, 2005.
2. S. C. Brenner, The mathematical theory of finite element methods, Springer, 2008.
3. I. Kissami, Manapy: An MPI-based Python framework for solving Poisson's equation using finite volume on unstructured-grid, in: AIP Conference Proceedings, Vol. 3034, AIP Publishing, 2024.
4. M. J. Gander, S. Loisel, D. B. Szyld, An optimal block iterative method and preconditioner for banded matrices with applications to PDEs on irregular domains, *SIAM Journal on Matrix Analysis and Applications* 33 (2) (2012) 653–680.
5. L. L. J. Côté, M. J. Gander, S. Loisel, Comparison of the Dirichlet-Neumann and Optimal Schwarz Method on the Sphere, in: *Domain Decomposition Methods in Science and Engineering XXVI*, Vol. 40, Springer, 2004, p. 235–242.
6. X.-C. Cai, M. Sarkis, A restricted additive Schwarz preconditioner for general sparse linear systems, *SIAM Journal on Scientific Computing* 21 (2) (1999) 792–797.
7. H. Liu, Z. Chen, S. Yu, B. Hsieh, L. Shao, Development of a Restricted Additive Schwarz Preconditioner for Sparse Linear Systems on NVIDIA GPU, *International Journal of Numerical Analysis and Modeling, Series B* 5 (1–2) (2014) 13–20.
8. Y. Osaki, K.-I. Ishikawa, Domain Decomposition Method on GPU Cluster, in: *Proceedings of the XXVIII International Symposium on Lattice Field Theory (Lattice 2010)*, 2010. arXiv:arXiv:1011.3318v1.
9. P. Nayak, T. Cojean, H. Anzt, Evaluating Asynchronous Schwarz Solvers on GPUs, *The International Journal of High Performance Computing Applications* 35 (3) (2021). doi:10.1177/1094342020946814.
10. A. St-Cyr, M. J. Gander, S. J. Thomas, Optimized multiplicative, additive, and restricted additive Schwarz preconditioning, *SIAM Journal on Scientific Computing* 29 (6) (2007) 2402–2425.
11. I. Kissami, Mumps4py: Python interface for the MUMPS solver (2025).
12. I. Kissami, R. Basmadjian, O. Chakir, M. R. Abid, TOUBKAL: a high-performance supercomputer powering scientific research in Africa, *The Journal of Supercomputing* 81 (15) (2025) 1–45.
13. J. Fořt, J. Karel, D. Trdlička, F. Benkhaldoun, I. Kissami, J. B. Montavon, J. Z. Mezei, Finite volume methods for numerical simulation of the discharge motion described by different physical models, Vol. 45, *Advances in Computational Mathematics*, 2019.
14. M. J. Gander, L. Laayouni, D. B. Szyld, Sparse approximate inverse (spai) based transmission conditions for optimized algebraic schwarz methods, in: *Domain Decomposition Methods in Science and Engineering XXVI*, Vol. 149, Springer, 2023, pp. 399–406.
15. M. J. Gander, L. Laayouni, D. B. Szyld, An Alternating Approach for Optimizing Transmission Conditions in Algebraic Schwarz Methods, in: *Domain Decomposition Methods in Science and Engineering XXVI*, Vol. 145, Springer, 2023, pp. 343–350.