Htool-DDM: A C++ library for parallel solvers and compressed linear systems.

Pierre Marchand ⓘ [1,¶], **Pierre-Henri Tournier**[2]**, and Pierre Jolivet**[3]

1 POEMS, CNRS, Inria, ENSTA, Institut Polytechnique de Paris, 91120 Palaiseau, France **2** Sorbonne Université, Université Paris Cité, CNRS, Inria, Laboratoire Jacques-Louis Lions, LJLL, EPC ALPINES, 4 place Jussieu, Paris F-75005, France **3** Sorbonne Université, CNRS, LIP6, 75252 Paris, France **¶** Corresponding author

Summary

Compressible dense linear systems arise in many applications such as:

- discretization of boundary integral equations (Börm et al., 2003),
- solving Lyapunov and Riccati equations (Börm et al., 2003),
- discretization of the integral Fractional Laplacian (Ainsworth & Glusa, 2018), and
- kernel-based scattered data interpolation (Iske et al., 2017).

They are often derived from a discretization of asymptotically smooth kernels $\kappa(\mathbf{x}, \mathbf{y})$, i.e., for $\mathbf{x} \neq \mathbf{y}$

$$|\partial_x^\alpha \partial_y^\beta \kappa(\mathbf{x}, \mathbf{y})| \leq C_{\mathrm{as}} |\mathbf{x} - \mathbf{y}|^{-|\alpha| - |\beta| - s},$$

where $\alpha, \beta \in \mathbb{N}_0^d$, with $\alpha + \beta \neq 0$ and $C_{\mathrm{as}}, s \in \mathbb{R}$ are constants.

This mathematical property implies that the interaction between two distant clusters can be represented by a low-rank matrix. Multiple compression techniques have been developed to take advantage of this low-rank structure and provide an approximated representation of the a priori dense linear systems, for example, $\mathcal{H}$-matrix and $\mathcal{H}^2$-matrix (Bebendorf, 2008; Börm, 2010; Hackbusch, 2015) and FMM (Greengard & Rokhlin, 1987). The advantages are usually:

- to avoid the assembly of the full dense linear system, and
- to provide a compressed/approximated version of all or part of the usual dense linear algebra.

The goal of such compression techniques is to lower the memory footprint and the cost of solving the linear system. For the latter, they provide an approximated matrix-vector product whose complexity scales linearly or quasi-linearly with the size of the problem. Thus, iterative solvers such as Conjugate Gradient or GMRES are well-suited.

Beside compression, another technique to accelerate iterative linear solvers is to use preconditioners: instead of solving $\mathbf{A}\mathbf{x} = \mathbf{f}$, where $\mathbf{A}$ is a dense or compressed linear system, we solve $\mathbf{P}\mathbf{A}\mathbf{x} = \mathbf{P}\mathbf{f}$ where $\mathbf{P}$ should be relatively inexpensive to apply, and $\mathbf{P}\mathbf{A}$ more suitable for an iterative resolution.

A class of preconditioners stemming from domain decomposition methods (DDM) are *Schwarz preconditioners*. They rely on a decomposition with overlap of the initial domain. $\mathbf{P}$ is constructed by solving well-chosen problems within each subdomain, together with a small global problem, called coarse space, that provides the necessary global information for the method to retain robustness as the number of subdomains increases. Such preconditioners have been introduced for boundary integral equations in Hebeker (1990) and analyzed in Stephan & Tran (1998) and Heuer (1996).

One goal of `Htool-DDM` is to provide black-box DDM solvers using an adaptive coarse space called GenEO for "Generalized Eigenproblems problem in the Overlap" (Spillane et al., 2013). It has been adapted for specific boundary integral problems in Marchand et al. (2020) and Marchand (2020). The library also includes a default black-box matrix compression via an in-house $\mathcal{H}$-matrix implementation. `Htool-DDM` is a flexible platform to use and develop new DDM preconditioners and explore compression techniques via its multiple customization points.
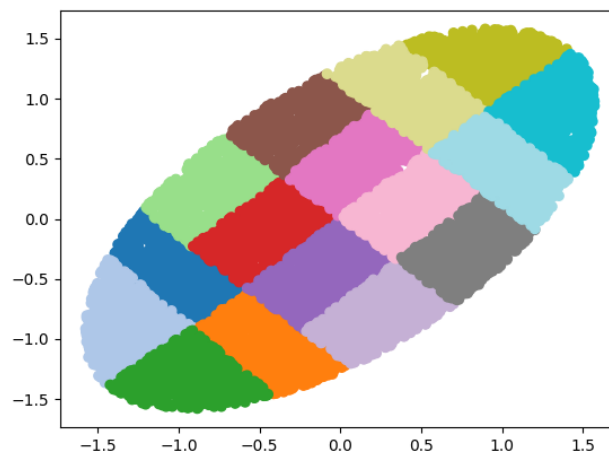
Statement of need

`Htool-DDM` is a lightweight C++ library that provides an easy-to-use interface for distributed iterative solvers and standard black-box matrix compression techniques via an in-house $\mathcal{H}$-matrix implementation. Its goal is to provide DDM preconditioners for dense/compressed linear systems.

It has many customization points to support research on efficient DDM preconditioners and compression techniques. For example, the user can provide their own compression algorithm, or customize the default hierarchical compression. Via its interface with HPDDM from Jolivet et al. (2013), it is also a flexible tool to test various iterative solvers and preconditioners, where local and/or global problems associated with DDM preconditioners can be tailored to the problem at hand.

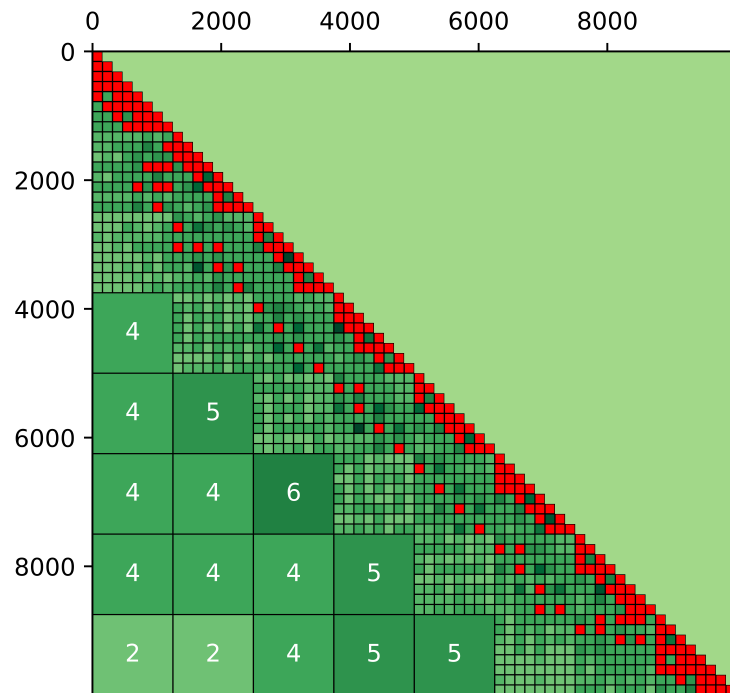The library has four main components:

- `Cluster` contains a hierarchical partition of a geometry. Various partition strategies are available, and users can define their own. See Figure 1 and examples/use_clustering.cpp.

Figure 1: Level 2 of cluster (quad)tree for rotated ellipse.

- `HMatrix` represents a compressed kernel using $\mathcal{H}$-matrix based on:
  - a user-defined function that takes row and column indexes and generates the associated subblock of the matrix to be compressed; and
  - cluster trees contained in `Cluster` objects and representing the hierarchical partition of the underlying geometry on which the kernel is applied.

Most linear algebra is supported with shared-memory parallelism and an interface inspired by BLAS/LAPACK and std::linalg. Basic compression and recompression techniques are available, and users can also provide their own. See Figure 2 and examples/use_hmatrix.cpp.

Figure 2: Symmetric $\mathcal{H}$-matrix for $\frac{1}{1e^{-5}+|x-y|}$. Red blocks are dense, numbers in green blocks correspond to rank.

- `DistributedOperator` is an MPI-distributed linear operator with a matrix-vector/matrix-matrix product. By default, it is defined as row-wise distributed operator where it uses locally a `HMatrix` for compression. It can also be interfaced with an external compression provider. See examples/use_distributed_operator.cpp.
- `DDM` models an iterative solver based on a `DistributedOperator` and HPDDM (Jolivet et al., 2013). It also deals with the assembly and application of DDM preconditioners. The latter can be built from local problems solved with $\mathcal{H}$-LU factorization, for example. See examples/use_ddm_solver.cpp.

Examples of libraries related to compression are `H2lib` (Börm, 2016), `hmat-oss` ("hmat-oss," 2024), `HODLRlib` (Ambikasaran et al., 2019), and `H2Opus` (Zampini et al., 2022). One major distinction of `Htool-DDM` from these libraries is its focus on DDM preconditioners, where the in-house compression is only one component that can actually be replaced or used in conjunction with such external libraries. It should also be noted that these libraries often provide the discretization of the problem at hand directly (typically a finite element method for boundary integral equations). This is not the case of `Htool-DDM` where the interface is kept algebraic.

The library has been used for the numerical experiments in Marchand et al. (2020) and Marchand (2020). It has been included in FreeFEM ((Hecht, 2012), version $\geq$ 4.5) to support boundary integral equations, and in PETSc ((Balay et al., 2025), version $\geq$ 3.16) for black-box compression and DDM solvers using the MatHtool PETSc matrix type and seamless integration into PCHPDDM (Jolivet et al., 2021). It also has its own Python interface.

We refer to the documentation for more details.

Acknowledgements

shaped the current version of `Htool-DDM`.

References

Ainsworth, M., & Glusa, C. (2018). Towards an efficient finite element method for the integral fractional Laplacian on polygonal domains. In J. Dick, F. Y. Kuo, & H. Woźniakowski (Eds.), *Contemporary computational mathematics - a celebration of the 80th birthday of Ian Sloan* (pp. 17–57). Springer International Publishing. https://doi.org/10.1007/978-3-319-72456-0_2

Ambikasaran, S., Singh, K., & Sankaran, S. (2019). HODLRlib: A library for hierarchical matrices. *Journal of Open Source Software*, *4*(34), 1167. https://doi.org/10.21105/joss.01167

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., … Zhang, J. (2025). *PETSc/TAO users manual* (ANL-21/39 - Revision 3.24). Argonne National Laboratory. https://doi.org/10.2172/2998643

Bebendorf, M. (2008). *Hierarchical matrices: A means to efficiently solve elliptic boundary value problems* (Vol. 63). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-77147-0

Börm, S. (2010). *Efficient numerical methods for non-local operators: $\mathcal{H}^2$-matrix compression, algorithms and analysis*. European Mathematical Society. https://doi.org/10.4171/091

Börm, S. (2016). H2lib. In *GitHub repository*. GitHub. https://github.com/H2Lib/H2Lib/tree/v3.0

Börm, S., Grasedyck, L., & Hackbusch, W. (2003). Introduction to hierarchical matrices with applications. *Engineering Analysis with Boundary Elements*, *27*(5), 405–422. https://doi.org/10.1016/S0955-7997(02)00152-2

Greengard, L., & Rokhlin, V. (1987). A fast algorithm for particle simulations. *Journal of Computational Physics*, *73*(2), 325–348. https://doi.org/10.1016/0021-9991(87)90140-9

Hackbusch, W. (2015). *Hierarchical matrices: Algorithms and analysis* (Vol. 49). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-47324-5

Hebeker, F. K. (1990). On a parallel Schwarz algorithm for symmetric strongly elliptic integral equations. In R. Glowinski, Y. A. Kuznetsov, G. Meurant, J. Périaux, & O. B. Widlund (Eds.), *Domain decomposition methods for partial differential equations* (pp. 382–393). SIAM.

Hecht, F. (2012). New development in FreeFem++. *Journal of Numerical Mathematics*, *20*(3-4), 251–265. https://doi.org/10.1515/jnum-2012-0013

Heuer, N. (1996). Efficient algorithms for the p-version of the boundary element method. *Journal of Integral Equations and Applications*, *8*(3), 337–361. https://doi.org/10.1216/jiea/1181075956

hmat-oss. (2024). In *GitHub repository*. GitHub. https://github.com/jeromerobert/hmat-oss

Iske, A., Le Borne, S., & Wende, M. (2017). Hierarchical matrix approximation for kernel-based scattered data interpolation. *SIAM Journal on Scientific Computing*, *39*(5), A2287–A2316. https://doi.org/10.1137/16m1101167

Jolivet, P., Hecht, F., Nataf, F., & Prudhomme, C. (2013). Scalable domain decomposition preconditioners for heterogeneous elliptic problems. *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, 1–11.

https://doi.org/10.1145/2503210.2503212

Jolivet, P., Roman, J. E., & Zampini, S. (2021). KSPHPDDM and PCHPDDM: Extending PETSc with advanced Krylov methods and robust multilevel overlapping Schwarz preconditioners. *Computers & Mathematics with Applications*, *84*, 277–295. https://doi.org/10.1016/j.camwa.2021.01.003

Marchand, P. (2020). *Schwarz methods and boundary integral equations* [Theses, Sorbonne Université]. https://hal.archives-ouvertes.fr/tel-02922455

Marchand, P., Claeys, X., Jolivet, P., Nataf, F., & Tournier, P.-H. (2020). Two-level preconditioning for h-version boundary element approximation of hypersingular operator with GenEO. *Numerische Mathematik*, *146*(3), 597–628. https://doi.org/10.1007/s00211-020-01149-5

Spillane, N., Dolean, V., Hauret, P., Nataf, F., Pechstein, C., & Scheichl, R. (2013). Abstract robust coarse spaces for systems of PDEs via generalized eigenproblems in the overlaps. *Numerische Mathematik*, *126*(4), 741–770. https://doi.org/10.1007/s00211-013-0576-y

Stephan, E. P., & Tran, T. (1998). Domain decomposition algorithms for indefinite hypersingular integral equations: The h and p versions. *SIAM Journal on Scientific Computing*, *19*(4), 1139–1153. https://doi.org/10.1137/S1064827595296161

Zampini, S., Boukaram, W., Turkiyyah, G., Knio, O., & Keyes, D. (2022). H2Opus: A distributed-memory multi-GPU software package for non-local operators. *Advances in Computational Mathematics*, *48*(31). https://doi.org/10.1007/s10444-022-09942-6