

Journal Pre-proof

Parallel Sparse Computation Toolkit

Pasqua D'Ambra, Fabio Durastante, Salvatore Filippone

PII: S2665-9638(22)00147-6
DOI: <https://doi.org/10.1016/j.simpa.2022.100463>
Reference: SIMPA 100463

To appear in: *Software Impacts*

Received date: 30 November 2022
Revised date: 21 December 2022
Accepted date: 28 December 2022



Please cite this article as: P. D'Ambra, F. Durastante and S. Filippone, Parallel Sparse Computation Toolkit, *Software Impacts* (2023), doi: <https://doi.org/10.1016/j.simpa.2022.100463>.

This is a PDF file of an article that has undergone enhancements after acceptance, such as the addition of a cover page and metadata, and formatting for readability, but it is not yet the definitive version of record. This version will undergo additional copyediting, typesetting and review before it is published in its final form, but we are providing this version to give early visibility of the article. Please note that, during the production process, errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

© 2022 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

*Parallel Sparse Computation Toolkit**Pasqua D'Ambra¹, Fabio Durastante²¹, Salvatore Filippone³¹***Abstract**

This paper presents a new software framework for solving large and sparse linear systems on current hybrid architectures, from small servers to high-end supercomputers, embedding multi-core CPUs and Nvidia GPUs at the node level. The framework has a modular structure and is composed of three main components, which separate basic functionalities for managing distributed sparse matrices and executing some sparse matrix computations involved in iterative Krylov projection methods, eventually exploiting multi-threading and CUDA-based programming models, from the functionalities for setup and application of different types of one-level and multi-level algebraic preconditioners.

Keywords

linear solvers, algebraic preconditioners, HPC, GPU, heterogeneous computing

Code metadata

Nr.	Code metadata description	Please fill in this column
C1	Current code version	v1.0 (PSBLAS v3.8.0-2, AMG4PSBLAS v1.1.0, PSBLAS3-EXT v1.3.0.1)
C2	Permanent link to code/repository used for this code version	https://github.com/psctoolkit/psctoolkit
C3	Permanent link to Reproducible Capsule	https://doi.org/10.24433/CO.0851253.v1
C4	Legal Code License	BSD 3.0
C5	Code versioning system used	git
C6	Software code languages, tools, and services used	Fortran, C, C++, MPI, CUDA, OpenMP
C7	Compilation requirements, operating environments & dependencies	GNU Toolchain (v. $\geq 6.0.0$), OpenMPI or MPICH or Spectrum MPI with OpenBLAS or ATLAS or BLAS/LAPACK.
C8	If available Link to developer documentation/manual	PSBLAS https://psctoolkit.github.io/psblasguide/psblas-3.8.pdf ; AMG4PSBLAS https://psctoolkit.github.io/amg4psblasguide/amg4psblas_1.0-guide.pdf ; PSBLAS-EXT https://github.com/sfilippone/psblas3-ext/raw/maint-1.3/docs/psblas-ext-1.3.pdf
C9	Support email for questions	psctoolkit@na.iac.cnr.it

¹Institute for Applied Computing “Mauro Picone”, National Research Council, Naples, Italy. E-mail pasqua.dambra@cnr.it.

²Dept. of Mathematics, University of Pisa, Pisa, Italy. E-mail fabio.durastante@unipi.it

³Dept. of Civil and Computer Engineering, University of Rome “Tor Vergata”, Rome, Italy. E-mail salvatore.filippone@uniroma2.it.

1. The Parallel Sparse Computation Toolkit

Sparse linear algebra is essential for a wide variety of scientific applications, from the solution of a discretized system of either linear or nonlinear partial differential equations to the analysis of large complex networks, or the computation of a few eigenvalues and eigenvectors of an operator. For all these problems having parallel sparse solvers is pivotal and lies at the core of the quasi-totality of multi-physics and multi-scale simulations. The technological challenge we face nowadays is investing effort and research in efficiently using hybrid architectures, embedding multi-core CPUs, and many-core accelerators, such as GPUs. These architectures are widely available in large infrastructures hosting petascale and pre-exascale platforms as well as in small data centers. The suite of libraries we present here tries to face the above challenge by offering a framework in which it is one hand possible to use scalable algorithmic solutions to make extreme scale computing possible, **on the other one hand** to leverage a set of tools for efficient exploitation of multi-core CPUs and GPU accelerators in sparse matrix computations at the node level.

The Parallel Sparse Computation Toolkit (PSCToolkit) is made of three main building blocks

PSBLAS was developed with the aim to facilitate the parallelization of computationally intensive scientific applications and is designed to address the parallel implementation of iterative solvers for sparse linear systems through the distributed-memory paradigm with message passing [26, 27]. It includes routines for multiplying sparse matrices by dense matrices, solving block diagonal systems with triangular diagonal entries, and preprocessing sparse matrices. It contains also additional routines for dense matrix operations. In order to address the solution of extreme-scale problems, in the last versions of PSBLAS, effective handling of large index spaces requiring 8-byte integers has been implemented.

AMG4PSBLAS is a package of parallel algebraic preconditioners. It is a progress of a software development project started in 2007, named MLD2P4 [15, 16, 21], which originally implemented a multilevel version of some domain decomposition preconditioners of additive-Schwarz type, and was based on a parallel decoupled version of the well known smoothed aggregation method [14, 41] to generate multilevel hierarchies of coarser matrices. The current version of the library extends the previous one by including many new smoothers and more general multigrid cycles and relies on a parallel implementation of the Coarsening based on Compatible Weighted Matching introduced in [20, 23] and described in detail in [18].

PSBLAS-EXT contains a set of extensions to the PSBLAS library. The extensions provide additional storage matrix formats beyond the ones already contained in PSBLAS [17, 26], as well as interfaces to two external libraries called SPGPU, for computations on NVIDIA GPUs and LIBRSB [33], for computations on multi-core parallel machines.

The PSCToolkit is implemented in the Fortran 2003 [34] programming language, with reuse and/or adaptation of existing Fortran 77 and Fortran 95 software, plus a handful of C and C++ routines. At the user level, the library can be called from both Fortran and C/C++. In Figure 1 we give a graphical depiction of the whole structure.

2. Impact and use cases

The prototypical problem that PSCToolkit permits to solve is the solution of a large linear system of the form

$$A\mathbf{x} = \mathbf{b}, \quad A \in \mathbb{K}^{n \times n}, \quad \mathbf{x}, \mathbf{b} \in \mathbb{K}^n, \quad \mathbb{K} = \{\mathbb{R}, \mathbb{C}\}, \quad (1)$$

with A a *sparse matrix*, i.e., a matrix with a number of *nonzero entries* of the order of n , by using a Krylov method [39] – CG [28], FCG [35], GMRES [37], BiCGStab [43], BiCGStab(ℓ) [40] – in conjunction with a purely algebraic **preconditioner** P transforming (1) into an equivalent one

$$P^{-1}A\mathbf{x} = P^{-1}\mathbf{b}, \quad \text{or} \quad AP^{-1}(P\mathbf{x}) = \mathbf{b}, \quad \text{or} \quad P_1^{-1}AP_2^{-1}(P_2\mathbf{x}) = P_1^{-1}\mathbf{b} \text{ with } P = P_1P_2,$$

with more favorable convergence properties. The toolkit permits to combine together classical stationary iterative methods – Jacobi, Gauss-Seidel, and their ℓ_1 and block-variants – together with incomplete matrix

factorizations [10, 12] – Incomplete LU factorizations [38], Approximate Inverses [6–8], sparse inverse LU factors [25] – and domain decomposition and algebraic multilevel strategies – Additive Schwarz, Restricted Additive Schwarz, Decoupled Smoothed Aggregation [14, 41], Parallel Coupled smoothed and unsmoothed aggregation based on Compatible Weighted Matching [20, 23] with V -, W - and K -cycles [42].

The various libraries offer a simplicity of extension that allows users to leverage the distributed environment management tools on the one hand to directly solve the linear systems that come from their application, on the other hand, to try new algorithmic solutions focusing mostly on mathematical aspects while delegating to the library many of the bearings related to the parallel environment. Furthermore, the solution of linear systems is not the only problem that can be faced with the tools available in the toolkit. More generally, problems that can be described in terms of BLAS for sparse arrays in a distributed environment can be implemented coherently with these tools, consider, e.g., the computation of eigenpairs, the computation of matrix functions, the solution of matrix equations using projection methods and various model order reduction techniques employing Krylov subspaces methods.

In this spirit, within the EoCoE (Energy-oriented Center of Excellence: towards Exascale for Energy) European project, the `PSCToolkit` has been recently interfaced and integrated into the `Alya` code [36], this is a simulation code for high-performance computational mechanics, which is part of the Unified European Applications Benchmark Suite (UEBAS) of scalable, currently relevant and publicly available application codes and datasets, of a size which can realistically be run on large systems, and maintained into the future. It enables the solution of coupled multiphysics problems using distributed and shared memory supercomputers and now permits the direct use of linear solvers and preconditioners from `PSCToolkit`. To test the use of the library on some example problems in an HPC environment it is available the Software as a Service (SaaS) portal provided by the Poznan Supercomputing and Networking Center (PSNC) at the URL eocoe.psnc.pl. Furthermore, some interfacing modules for the KINSOL package available from the Sundials project [29] have been developed, that enable KINSOL to use the solvers and preconditioners from `PSCToolkit` inside its Newton-based non-linear procedure. These modules and the documentation which details their implementations are available from the GitHub repository. The interfacing modules extend the functionalities of `PSCToolkit` for dealing with algebraic non-linear systems and impacts on KINSOL so that it can leverage linear solvers made available by the toolkit.

2.1 An example of use

To demonstrate the applicability of the library and its possibility of use on different machines, we consider one of the test examples distributed with `AMG4PSBLAS` using different combinations of architectures and development environments; see Table 1. The test relies on the classic Poisson problem with homogeneous Dirichlet boundary conditions:

$$\begin{cases} \nabla^2 u(\mathbf{x}) = f(\mathbf{x}), & \mathbf{x} \in \Omega = [0, 1]^3, \\ u(\mathbf{x}) = 0, & \mathbf{x} \in \partial\Omega, \end{cases}$$

discretized by the 7-points finite-differences method on the homogeneous tensor mesh. This discretization generates a symmetric positive definite and ill-conditioned linear system. For its solution, we choose to use the flexible conjugate gradient (FCG) method to achieve a residual on the error of 10^{-6} preconditioned by one of the algebraic multigrid preconditioners available in `AMG4PSBLAS`. Specifically, we consider a single sweep of V-cycle leveraging an AMG hierarchy with a smoothed prolongator obtained by the Coarsening based on Compatible Weighted Matching from [18] with aggregates of size at most 8. We use 4 sweeps of ℓ_1 -Jacobi [5] both as pre- and post-smoother. On the coarsest grid, we use again a Krylov method preconditioned by a block-Jacobi method with incomplete LU factorization on the local blocks. In Figure 2a we report the total solve time for a *weak-scaling* test using the MPI functionalities of the library, i.e., we show how the solution time varies with the number of computing units for a fixed problem size per unit – here fixed to approximately 256000 unknowns.

In the second test, we look again at the same problem and consider the same algorithmic configuration as above for the AMG preconditioner, except for the coarsest solver, where we use 30 iterations of the ℓ_1 -Jacobi method. In this case, we compare a sequential run for the whole reachable size in terms of unknowns, versus a pure MPI run using the full CPU capabilities of the node, and a run using the available GPU accelerators on the given node; Figure 2b. On the *Workstation 2* (Table 1) we obtain for the GPU run a speedup of $21\times$ with respect to the sequential solve time, and of $5\times$ with respect to the solve time when the full CPU is used

(40 MPI tasks). On the *Large Cluster*, which has 4 GPUs per node, we obtain a speedup of $68\times$ with respect to the sequential solve time, and – consistently – of $5\times$ with respect to the solve time when the full CPU is used (32 MPI tasks). Finally, in Figure 2c we consider again a weak scaling on the *Large Cluster* fully using the computing nodes and around 8192000 unknowns per node. That is, we compare the scaling using either 32 MPI tasks per computing node or 4 MPI tasks and 4 GPUs per node while using up to 16 nodes. We observe behavior in line with the expected one given the performances on a single node and the performances using pure MPI. Small fluctuations are observed in the solution times which seem mostly attributable to the machine; in fact, the number of iterations for the use case with pure MPI is contained between 8 and 15, while in the hybrid case between 8 and 12.

For results in extreme scalability and comparison tests between the various preconditioners and some competing libraries, we refer to [18].

3. Ongoing projects

The PSCToolkit is being used and going under active development in the Euro-HPC project “TEXTAROSSA: Towards EXtreme scale Technologies and Accelerators for euROhpc hw/Sw Supercomputing Applications for exascale.”

4. Limitations and further developments

Currently, support for OpenMP is limited to some internal functions. One of the development directions we are following is to extend it to heavier routines, such as the setup phase on the CPU of multigrid preconditioners to then exploit the application on the GPU. The other two lines of development underway are on one hand the use of communication-avoiding Krylov methods to reduce the weight of the scalar products and (re)orthogonalization steps on the total cost of the method. On the other, we plan to investigate the usage of hierarchical and other data-sparse matrix structure for reducing the memory footprint of the various methods.

5. Scholarly publications enabled by PSCToolkit

PSCToolkit has been used in several contexts on a range of different architectures and software stacks. Applications are the construction of relaxation method for high-performance image segmentation on GPUs [19]; the solution of the pressure equation in the Large-Eddy-Simulation of the Navier-Stokes equations [3, 4, 22, 36]; the numerical solution of discretized elliptic differential equations [13, 18]; the numerical solution of the linear systems arising from the application of a Quasi-Newton method to a discrete version of the Richards equation for the filtration of fluids in the vadose zone [9]. Some of the tools are discussed in the book [11] where parallel AMG methods for solving linear systems are illustrated.

Acknowledgements

This work was partially supported by GNCS-INdAM and by the Euro-HPC project *Towards EXtreme scale Technologies and Accelerators for euROhpc hw/Sw Supercomputing Applications for exascale (TEXTAROSSA)*, Horizon 2020 Program for Research and Innovation, ID: 956831. The authors are members of the INdAM research group GNCS.

References

- [1] P. Amestoy, A. Buttari, I. Duff, A. Guermouche, J.-Y. L'Excellent, and B. Uçar. “Mumps”. In: *Encyclopedia of Parallel Computing*. Ed. by D. Padua. Boston, MA: Springer US, 2011, pp. 1232–1238. ISBN: 978-0-387-09766-4. DOI: [10.1007/978-0-387-09766-4_204](https://doi.org/10.1007/978-0-387-09766-4_204). URL: https://doi.org/10.1007/978-0-387-09766-4_204.
- [2] P. R. Amestoy, T. A. Davis, and I. S. Duff. “Algorithm 837: AMD, an approximate minimum degree ordering algorithm”. In: *ACM Trans. Math. Software* 30.3 (2004), pp. 381–388. ISSN: 0098-3500. DOI: [10.1145/1024074.1024081](https://doi.org/10.1145/1024074.1024081). URL: <https://doi.org/10.1145/1024074.1024081>.
- [3] A. Aproxitola, P. D'Ambra, F. Denaro, D. di Serafino, and S. Filippone. “Scalable algebraic multilevel preconditioners with application to CFD”. In: *Parallel computational fluid dynamics 2008*. Vol. 74. Lect. Notes Comput. Sci. Eng. Springer, Heidelberg, 2010, pp. 15–27. DOI: [10.1007/978-3-642-14438-7_2](https://doi.org/10.1007/978-3-642-14438-7_2). URL: https://doi.org/10.1007/978-3-642-14438-7_2.
- [4] A. Aproxitola, P. D'Ambra, F. M. Denaro, D. di Serafino, and S. Filippone. “SPaC-LES: enabling large eddy simulations with parallel sparse matrix computation tools”. In: *Comput. Math. Appl.* 70.11 (2015), pp. 2688–2700. ISSN: 0898-1221. DOI: [10.1016/j.camwa.2015.06.028](https://doi.org/10.1016/j.camwa.2015.06.028). URL: <https://doi.org/10.1016/j.camwa.2015.06.028>.
- [5] A. H. Baker, R. D. Falgout, T. V. Kolev, and U. M. Yang. “Multigrid smoothers for ultraparallel computing”. In: *SIAM J. Sci. Comput.* 33.5 (2011), pp. 2864–2887. ISSN: 1064-8275. DOI: [10.1137/100798806](https://doi.org/10.1137/100798806). URL: <https://doi.org/10.1137/100798806>.
- [6] M. Benzi, J. K. Cullum, and M. Tũma. “Robust approximate inverse preconditioning for the conjugate gradient method”. In: *SIAM J. Sci. Comput.* 22.4 (2000), pp. 1318–1332. ISSN: 1064-8275. DOI: [10.1137/S1064827599356900](https://doi.org/10.1137/S1064827599356900). URL: <https://doi.org/10.1137/S1064827599356900>.
- [7] M. Benzi, C. D. Meyer, and M. Tũma. “A sparse approximate inverse preconditioner for the conjugate gradient method”. In: *SIAM J. Sci. Comput.* 17.5 (1996), pp. 1135–1149. ISSN: 1064-8275. DOI: [10.1137/S1064827594271421](https://doi.org/10.1137/S1064827594271421). URL: <https://doi.org/10.1137/S1064827594271421>.
- [8] M. Benzi and M. Tũma. “A sparse approximate inverse preconditioner for nonsymmetric linear systems”. In: *SIAM J. Sci. Comput.* 19.3 (1998), pp. 968–994. ISSN: 1064-8275. DOI: [10.1137/S1064827595294691](https://doi.org/10.1137/S1064827595294691). URL: <https://doi.org/10.1137/S1064827595294691>.
- [9] D. Bertaccini, P. D'Ambra, F. Durastante, and S. Filippone. *Why diffusion-based preconditioning of Richards equation works: spectral analysis and computational experiments at very large scale*. 2021. DOI: [10.48550/ARXIV.2112.05051](https://arxiv.org/abs/2112.05051). URL: <https://arxiv.org/abs/2112.05051>.
- [10] D. Bertaccini and F. Durastante. “Interpolating preconditioners for the solution of sequence of linear systems”. In: *Comput. Math. Appl.* 72.4 (2016), pp. 1118–1130. ISSN: 0898-1221. DOI: [10.1016/j.camwa.2016.06.023](https://doi.org/10.1016/j.camwa.2016.06.023). URL: <https://doi.org/10.1016/j.camwa.2016.06.023>.
- [11] D. Bertaccini and F. Durastante. *Iterative methods and preconditioning for large and sparse linear systems with applications*. Monographs and Research Notes in Mathematics. CRC Press, Boca Raton, FL, 2018, pp. xviii+353. ISBN: 978-1-4987-6416-2.
- [12] D. Bertaccini and S. Filippone. “Sparse approximate inverse preconditioners on high performance GPU platforms”. In: *Comput. Math. Appl.* 71.3 (2016), pp. 693–711. ISSN: 0898-1221. DOI: [10.1016/j.camwa.2015.12.008](https://doi.org/10.1016/j.camwa.2015.12.008). URL: <https://doi.org/10.1016/j.camwa.2015.12.008>.
- [13] A. Borzì, V. De Simone, and D. di Serafino. “Parallel algebraic multilevel Schwarz preconditioners for a class of elliptic PDE systems”. In: *Comput. Vis. Sci.* 16.1 (2013), pp. 1–14. ISSN: 1432-9360. DOI: [10.1007/s00791-014-0220-0](https://doi.org/10.1007/s00791-014-0220-0). URL: <https://doi.org/10.1007/s00791-014-0220-0>.
- [14] M. Brezina and P. Vaněk. “A black-box iterative solver based on a two-level Schwarz method”. In: *Computing* 63.3 (1999), pp. 233–263. ISSN: 0010-485X. DOI: [10.1007/s006070050033](https://doi.org/10.1007/s006070050033). URL: <https://doi.org/10.1007/s006070050033>.

- [15] A. Buttari, P. D'Ambra, D. di Serafino, and S. Filippone. "2LEV-D2P4: a package of high-performance preconditioners for scientific and engineering applications". In: *Appl. Algebra Engrg. Comm. Comput.* 18.3 (2007), pp. 223–239. ISSN: 0938-1279. DOI: [10.1007/s00200-007-0035-z](https://doi.org/10.1007/s00200-007-0035-z). URL: <https://doi.org/10.1007/s00200-007-0035-z>.
- [16] A. Buttari, P. D'Ambra, D. di Serafino, and S. Filippone. "Extending PSBLAS to Build Parallel Schwarz Preconditioners". In: *Applied Parallel Computing. State of the Art in Scientific Computing*. Ed. by J. Dongarra, K. Madsen, and J. Waśniewski. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 593–602. ISBN: 978-3-540-33498-9.
- [17] V. Cardellini, S. Filippone, and D. W. Rouson. "Design patterns for sparse-matrix computations on hybrid CPU/GPU platforms". In: *Scientific Programming* 22 (2014). 1, pp. 1–19. DOI: [10.3233/SPR-130363](https://doi.org/10.3233/SPR-130363). URL: <https://doi.org/10.3233/SPR-130363>.
- [18] P. D'Ambra, F. Durastante, and S. Filippone. "AMG preconditioners for linear solvers towards extreme scale". In: *SIAM J. Sci. Comput.* 43.5 (2021), S679–S703. ISSN: 1064-8275. DOI: [10.1137/20M134914X](https://doi.org/10.1137/20M134914X). URL: <https://doi.org/10.1137/20M134914X>.
- [19] P. D'Ambra and S. Filippone. "A parallel generalized relaxation method for high-performance image segmentation on GPUs". In: *J. Comput. Appl. Math.* 293 (2016), pp. 35–44. ISSN: 0377-0427. DOI: [10.1016/j.cam.2015.04.035](https://doi.org/10.1016/j.cam.2015.04.035). URL: <https://doi.org/10.1016/j.cam.2015.04.035>.
- [20] P. D'Ambra, S. Filippone, and P. S. Vassilevski. "BootCMatch: a software package for bootstrap AMG based on graph weighted matching". In: *ACM Trans. Math. Software* 44.4 (2018), Art. 39, 25. ISSN: 0098-3500. DOI: [10.1145/3190647](https://doi.org/10.1145/3190647). URL: <https://doi.org/10.1145/3190647>.
- [21] P. D'Ambra, D. di Serafino, and S. Filippone. "MLD2P4: a package of parallel algebraic multilevel domain decomposition preconditioners in Fortran 95". In: *ACM Trans. Math. Software* 37.3 (2010), Art. 30, 23. ISSN: 0098-3500. DOI: [10.1145/1824801.1824808](https://doi.org/10.1145/1824801.1824808). URL: <https://doi.org/10.1145/1824801.1824808>.
- [22] P. D'Ambra, D. di Serafino, and S. Filippone. "Performance analysis of parallel Schwarz preconditioners in the LES of turbulent channel flows". In: *Comput. Math. Appl.* 65.3 (2013), pp. 352–361. ISSN: 0898-1221. DOI: [10.1016/j.camwa.2012.06.023](https://doi.org/10.1016/j.camwa.2012.06.023). URL: <https://doi.org/10.1016/j.camwa.2012.06.023>.
- [23] P. D'Ambra and P. S. Vassilevski. "Adaptive AMG with coarsening based on compatible weighted matching". In: *Comput. Vis. Sci.* 16.2 (2013), pp. 59–76. ISSN: 1432-9360. DOI: [10.1007/s00791-014-0224-9](https://doi.org/10.1007/s00791-014-0224-9). URL: <https://doi.org/10.1007/s00791-014-0224-9>.
- [24] T. A. Davis. "Algorithm 832: UMFPACK V4.3—an unsymmetric-pattern multifrontal method". In: *ACM Trans. Math. Software* 30.2 (2004), pp. 196–199. ISSN: 0098-3500. DOI: [10.1145/992200.992206](https://doi.org/10.1145/992200.992206). URL: <https://doi.org/10.1145/992200.992206>.
- [25] A. C. N. van Duin. "Scalable parallel preconditioning with the sparse approximate inverse of triangular matrices". In: vol. 20. 4. Sparse and structured matrices and their applications (Coeur d'Alene, ID, 1996). 1999, pp. 987–1006. DOI: [10.1137/S0895479897317788](https://doi.org/10.1137/S0895479897317788). URL: <https://doi.org/10.1137/S0895479897317788>.
- [26] S. Filippone and A. Buttari. "Object-Oriented Techniques for Sparse Matrix Computations in Fortran 2003". In: *ACM Trans. Math. Softw.* 38.4 (Aug. 2012). ISSN: 0098-3500. DOI: [10.1145/2331130.2331131](https://doi.org/10.1145/2331130.2331131). URL: <https://doi.org/10.1145/2331130.2331131>.
- [27] S. Filippone and M. Colajanni. "PSBLAS: A Library for Parallel Linear Algebra Computation on Sparse Matrices". In: *ACM Trans. Math. Softw.* 26.4 (Dec. 2000), pp. 527–550. ISSN: 0098-3500. DOI: [10.1145/365723.365732](https://doi.org/10.1145/365723.365732). URL: <https://doi.org/10.1145/365723.365732>.
- [28] M. R. Hestenes and E. Stiefel. "Methods of conjugate gradients for solving linear systems". In: *J. Research Nat. Bur. Standards* 49 (1952), 409–436 (1953). ISSN: 0160-1741.
- [29] A. C. Hindmarsh, P. N. Brown, K. E. Grant, S. L. Lee, R. Serban, D. E. Shumaker, and C. S. Woodward. "SUNDIALS: Suite of Nonlinear and Differential/Algebraic Equation Solvers". In: *ACM Trans. Math. Softw.* 31.3 (Sept. 2005), pp. 363–396. ISSN: 0098-3500. DOI: [10.1145/1089014.1089020](https://doi.org/10.1145/1089014.1089020). URL: <https://doi.org/10.1145/1089014.1089020>.

- [30] G. Karypis and V. Kumar. “A fast and high quality multilevel scheme for partitioning irregular graphs”. In: *SIAM J. Sci. Comput.* 20.1 (1998), pp. 359–392. ISSN: 1064-8275. DOI: [10.1137/S1064827595287997](https://doi.org/10.1137/S1064827595287997). URL: <https://doi.org/10.1137/S1064827595287997>.
- [31] X. S. Li. “An Overview of SuperLU: Algorithms, Implementation, and User Interface”. In: *ACM Trans. Mathematical Software* 31.3 (Sept. 2005), pp. 302–325.
- [32] X. S. Li and J. W. Demmel. “SuperLU_DIST: A Scalable Distributed-Memory Sparse Direct Solver for Unsymmetric Linear Systems”. In: *ACM Trans. Mathematical Software* 29.2 (June 2003), pp. 110–140.
- [33] M. Martone, S. Filippone, S. Tucci, M. Paprzycki, and M. Ganzha. “Utilizing Recursive Storage in Sparse Matrix-Vector Multiplication - Preliminary Considerations”. In: *Proceedings of the ISCA 25th International Conference on Computers and Their Applications, CATA 2010, March 24-26, 2010, Sheraton Waikiki Hotel, Honolulu, Hawaii, USA*. Ed. by T. Philips. ISCA, 2010, pp. 300–305.
- [34] M. Metcalf, J. K. Reid, and M. Cohen. *Fortran 95/2003 Explained*. Vol. 3. Oxford University Press Oxford, 2004.
- [35] Y. Notay. “Flexible conjugate gradients”. In: *SIAM J. Sci. Comput.* 22.4 (2000), pp. 1444–1460. ISSN: 1064-8275. DOI: [10.1137/S1064827599362314](https://doi.org/10.1137/S1064827599362314). URL: <https://doi.org/10.1137/S1064827599362314>.
- [36] H. Owen, O. Lehmkuhl, P. D’Ambra, F. Durastante, and S. Filippone. *Alya towards Exascale: Algorithmic Scalability using PSCToolkit*. 2022. DOI: [10.48550/ARXIV.2210.16660](https://arxiv.org/abs/2210.16660). URL: <https://arxiv.org/abs/2210.16660>.
- [37] Y. Saad and M. H. Schultz. “GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems”. In: *SIAM J. Sci. Statist. Comput.* 7.3 (1986), pp. 856–869. ISSN: 0196-5204. DOI: [10.1137/0907058](https://doi.org/10.1137/0907058). URL: <https://doi.org/10.1137/0907058>.
- [38] Y. Saad. “ILUT: a dual threshold incomplete LU factorization”. In: *Numer. Linear Algebra Appl.* 1.4 (1994), pp. 387–402. ISSN: 1070-5325. DOI: [10.1002/nla.1680010405](https://doi.org/10.1002/nla.1680010405). URL: <https://doi.org/10.1002/nla.1680010405>.
- [39] Y. Saad. *Iterative methods for sparse linear systems*. Second. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2003, pp. xviii+528. ISBN: 0-89871-534-2. DOI: [10.1137/1.9780898718003](https://doi.org/10.1137/1.9780898718003). URL: <https://doi.org/10.1137/1.9780898718003>.
- [40] G. L. G. Sleijpen, H. A. van der Vorst, and D. R. Fokkema. “BiCGstab(l) and other hybrid Bi-CG methods”. In: *Numer. Algorithms* 7.1 (1994), pp. 75–109. ISSN: 1017-1398. DOI: [10.1007/BF02141261](https://doi.org/10.1007/BF02141261). URL: <https://doi.org/10.1007/BF02141261>.
- [41] P. Vaněk, J. Mandel, and M. Brezina. “Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems”. In: vol. 56. 3. International GAMM-Workshop on Multi-level Methods (Meisdorf, 1994). 1996, pp. 179–196. DOI: [10.1007/BF02238511](https://doi.org/10.1007/BF02238511). URL: <https://doi.org/10.1007/BF02238511>.
- [42] P. Vassilevski. *Multilevel block factorization preconditioners: matrix-based analysis and algorithms for solving finite element equations*. New York, USA: Springer, 2008.
- [43] H. A. van der Vorst. “Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems”. In: *SIAM J. Sci. Statist. Comput.* 13.2 (1992), pp. 631–644. ISSN: 0196-5204. DOI: [10.1137/0913035](https://doi.org/10.1137/0913035). URL: <https://doi.org/10.1137/0913035>.

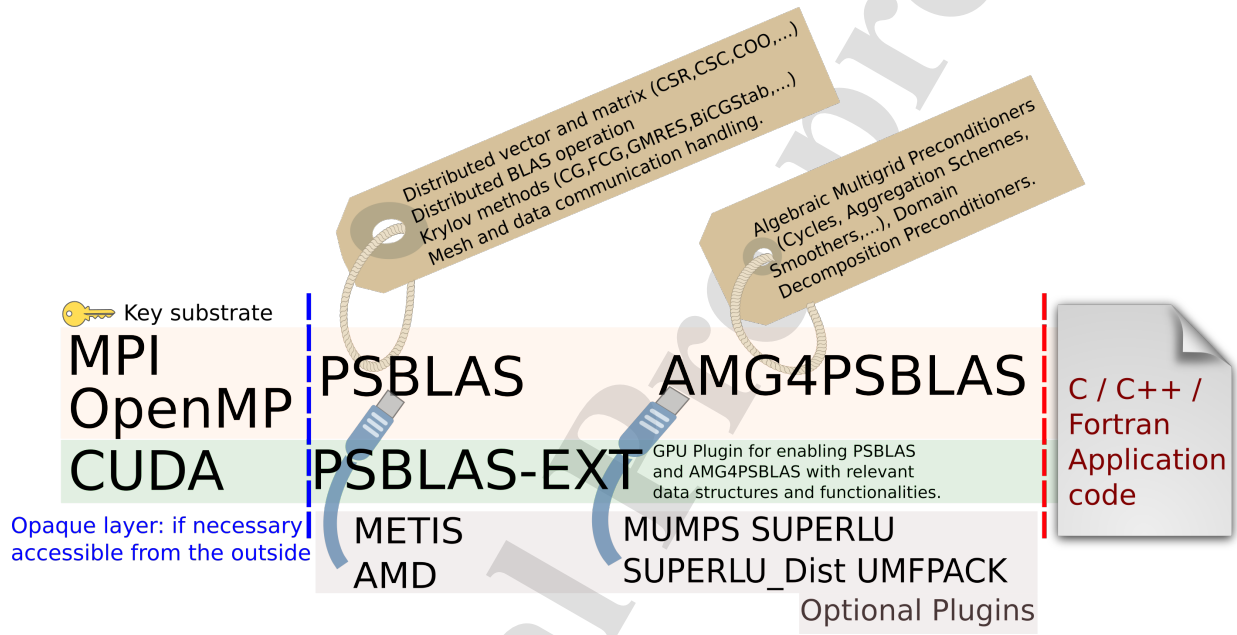
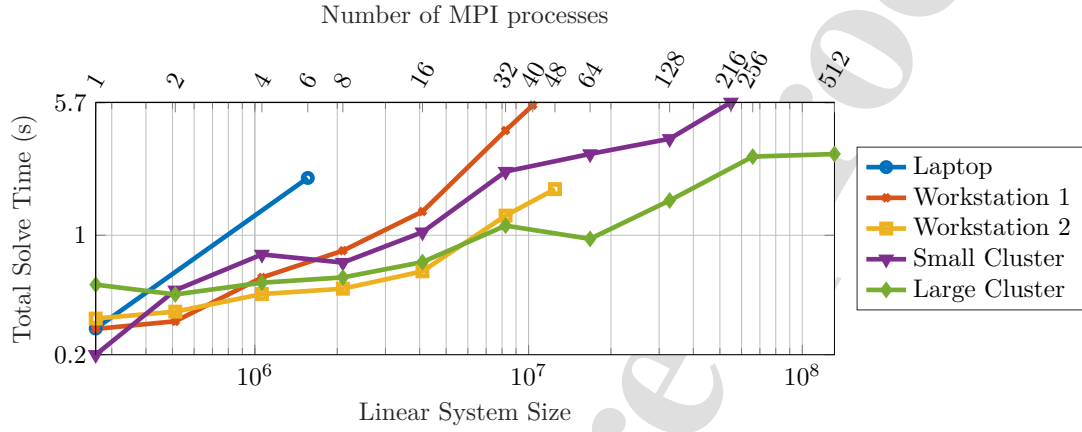


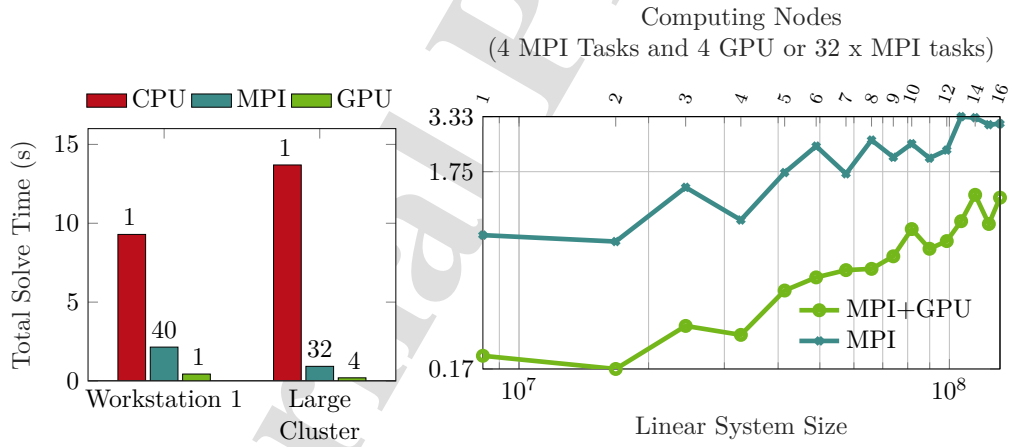
Figure 1: A graphical depiction of the structure of the library. The base layer is represented by an implementation of MPI (we have run against several open and proprietary implementations: OpenMPI, MPICH, Spectrum MPI, Intel oneAPI), OpenMP, and a version of the CUDA Compilation Toolkit to run on NVIDIA GPUs. Optional functionalities can be added from third-party libraries such as the Metis graph partitioner [30], SuiteSparse <http://suitesparse.com> (AMD [2], UMFPACK [24]), MUMPS [1], SuperLU [31] and SuperLU_dist [32]. The functionalities introduced by the toolkit (mesh and data handling, distributed data structure for vector and matrices, Krylov method, purely algebraic preconditioners) can be accessed by code written in both Fortran, C, and C++. In cases of a more elaborate integration with pre-existing code that needs to directly access the MPI/OpenMP/CUDA substrate, the toolkit is provided with routines to do so.

Machine	Hardware	Environment
Laptop	CPU: Intel ^(R) Core ^(TM) i7-8750H CPU @ 2.20GHz; Memory: 16Gb.	Compiler: GNU Suite 11.3.0; MPI: OpenMPI 4.1.2; BLAS: OpenBLAS 0.3.20.
Workstation 1	CPU: Intel ^(R) Xeon ^(R) Silver 4210 CPU @ 2.20GHz; Memory: 64Gb; GPU: NVIDIA Quadro RTX 5000.	Compiler: GNU Suite 11.3.0; MPI: mpich 3.4.3; CUDA: Cuda compilation tools, release 11.3, V11.3.109; BLAS: ATLAS 3.10.
Workstation 2	CPU: Intel ^(R) Xeon ^(R) Gold 6238R CPU @ 2.20GHz; Memory: 1.48T.	Compiler: GNU Suite 6.1.0; MPI: OpenMPI 1.10.7; BLAS: OpenBLAS 0.3.3.
Small Cluster	CPU: 1 Node (cl1) with Intel ^(R) Xeon ^(R) CPU E5-2643 v4 @ 3.40GHz, 4 nodes (cl2) with Intel ^(R) Xeon ^(R) CPU E5-2650 v4 @ 2.20GHz; Memory: (cl1) 126Gb, (cl2) 252Gb; Network: Intel 10-Gigabit X540-AT2.	Compiler: GNU Suite 12.2.0; MPI: OpenMPI 4.1.4; BLAS: OpenBLAS 0.3.20.
Large Cluster	CPU 980 nodes with 2×16 cores IBM POWER9 AC922 at 2.6(3.1) GHz; Memory: 256 GB/node; Network: Mellanox IB EDR DragonFly++ 100Gb/s; GPU: 4 × NVIDIA Volta V100 GPUs/node, Nvlink 2.0, 16GB.	Compiler: GNU Suite 11.2.0; MPI: OpenMPI 4.1.2; BLAS: BLAS 3.10.0; CUDA: Cuda compilation tools, release 11.6, V11.6.124.

Table 1: List of machines on which the default tests have been executed. The table shows the type of machine, the hardware used and the software environment in which the tests were performed. The *Large Cluster* is the Marconi-100 system by CINECA, the 24th machine in the TOP500 November 2022 list: <https://www.top500.org/lists/top500/list/2022/11/>.



(a) Weak scaling. The size per computing element is set to $n = 256000$ unknowns, and solve time is measured in seconds.



(b) Exploiting a full node, the number on top of the bars is the number of computing units.

(c) Weak scaling. The size per computing element is set to $n = 256000$ unknowns, and solve time is measured in seconds. We use from 1 to 16 nodes either with flat MPI (32 tasks per node) or using 4 MPI tasks per node with 4 GPUs.

Figure 2: Numerical examples run with the machines described in Table 1.

- software libraries for sparse linear algebra
- MPI/Cuda hybrid programming model for GPU exploitation
- iterative methods and algebraic preconditioners

Declaration of interests

☐ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐ The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.