New Coarse Corrections for Optimized Restricted Additive Schwarz Using PETSc

Martin J. Gander and Serge Van Criekingen

1 Introduction

Additive Schwarz Methods (ASM) are implemented in the PETSc library [2, 1, 3] within its `PCASM` preconditioning option. By default this applies the Restricted Additive Schwarz (RAS) method of Cai and Sarkis [4]. We here present the implementation, using PETSc tools, of two further improvements for this method: a new and more effective coarse correction, as well as optimized transmission conditions, resulting in an Optimized two-level Restricted Additive Schwarz (or ORAS2) method.

It is well known that domain decomposition methods applied to elliptic problems need a coarse correction to be scalable, since without it, information is only transferred from each subdomain to its direct neighbors which makes the number of iterations grow with the number of subdomains; for exceptions, see [6, 7]. Scalability is achieved by introducing a coarse grid on which a reduced-size calculation is performed to compute a coarse correction at each iteration of the solution process, yielding a two-level method. Our choice of the coarse grid points follows the method introduced in [11]: the coarse grid points are chosen in 1D to be the extreme grid points of the non-overlapping subdomains used to define RAS, and for a rectangular decomposition in 2D, four coarse grid points are placed around each cross point of the non-overlapping decomposition. This choice of placing the coarse grid nodes leads to substantially faster convergence than the classical option of equally distributing the coarse grid points within each subdomain.

As for optimized transmission conditions, we consider Robin transmission conditions instead of the classical Dirichlet ones, i.e., a well-chosen combination of Dirichlet and Neumann values at subdomain interfaces such as to minimize the

Martin J. Gander
University of Geneva, e-mail: `martin.gander@unige.ch`

Serge Van Criekingen
CNRS/IDRIS e-mail: `serge.van.criekingen@idris.fr`

number of iterations. We follow here the method described in [8] which only requires modifying the diagonal entries of interface nodes in the subdomain matrices. A good choice of these diagonal entries leads to a much faster convergence of the associated domain decomposition method than using the standard diagonal entries from RAS.

We present weak scaling numerical results on a 2-D Laplace test case using up to 16384 CPU cores. Combining coarse correction and optimized transmission conditions, we obtain substantially improved computation times with the new optimized two-level RAS method which, despite a larger memory footprint, proves to be competitive with the multigrid library HYPRE (with the default options of the PETSc interface to this library).

2 Coarse Correction and the two-level RAS method

We consider the solution of $A\mathbf{x} = \mathbf{b}$ on a domain Ω decomposed into a set of possibly overlapping subdomains Ω_j. Introducing a restriction operator R_j onto each subdomain Ω_j, local matrices can be built as $A_j = R_j\, A\, R_j^T$. To obtain the Restricted Additive Schwarz (RAS) method, we need also to introduce a partition of Ω into non-overlapping subdomains $\tilde{\Omega}_j$, as well as the corresponding restriction operators $\tilde{R}_j$. Then, the RAS method is defined by the iterations [4]

$$\mathbf{x}^{n+1} = \mathbf{x}^n + \sum_{j=1}^{J} \tilde{R}_j^T\, A_j^{-1}\, R_j\, (\mathbf{b} - A\mathbf{x}^n). \tag{1}$$

The RAS method has the drawback of yielding a non-symmetric system even for symmetric problems, but was shown to converge faster than the Additive Schwarz method because it remedies its non-convergent behavior in the overlaps [9].

To obtain a two-level method through coarse correction, we introduce a restriction operator R_c to the coarse space, such that the coarse system matrix reads $A_c = R_c\, A\, R_c^T$. In turn, the two-level RAS method with multiplicative coarse correction (denoted RAS2 in what follows) can be written as

$$\mathbf{x}^{n+1/2} = \mathbf{x}^n + \sum_{j=1}^{J} \tilde{R}_j^T\, A_j^{-1}\, R_j\, (\mathbf{b} - A\mathbf{x}^n), \tag{2}$$

$$\mathbf{x}^{n+1} = \mathbf{x}^{n+1/2} + R_c^T\, A_c^{-1}\, R_c\, (\mathbf{b} - A\mathbf{x}^{n+1/2}). \tag{3}$$

The definition of the coarse space, that is, the choice of the coarse grid nodes, is critical to obtain an efficient two-level method. Two possible choices are shown in Fig. 1. Compared to the classical approach (circles), the new approach (squares) introduced in [11] shows superior performance since it resolves the residual location along the interfaces well (see also [8, 12]), and is therefore the choice made here (-we however compare the iteration counts for the two methods in Section 4). For the

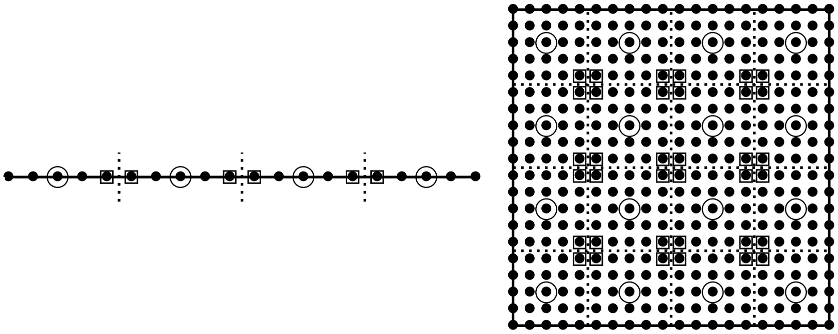


Fig. 1: Two choices of the coarse grid nodes in 1-D and 2-D: 1) the middle of each subdomain (circles) or 2) one node on each side of the (non-overlapping) subdomain interfaces (squares) in 1-D, or in 2-D four nodes around each cross point of the (non-overlapping) decomposition.

1-D case, it was actually shown in [11] that, for the Laplace equation, the new coarse correction yields convergence in two iterations, which is because the new coarse basis functions are harmonic within the subdomains.

In PETSc, the coarse correction was implemented using the PCSHELL preconditioning tool, which gives the possibility to entirely define the preconditioner. This self-defined preconditioner was then (multiplicatively) composed with the built-in PCASM (i.e., RAS) preconditioner using the PCCOMPOSITE composition tool [2].

3 Optimized Interface Conditions and the ORAS2 method

In the RAS iterations (1), each local A_j matrix corresponds to a discretized local problem with homogeneous Dirichlet boundary conditions. Optimized interface conditions are introduced by modifying these matrices into $\hat{A}_j$ matrices, each corresponding to a discretized local problem with homogeneous Robin boundary conditions of the type

$$\frac{\partial u_j}{\partial n_j} + p\, u_j = 0 \qquad \text{on } \partial\Omega_j \setminus \partial\Omega. \tag{4}$$

The resulting optimized RAS method will be denoted by ORAS, and a good choice of the parameter p in (4) is important for good performance.

Starting from the RAS2 iterations (2)-(3), the optimized two-level RAS method, denoted by ORAS2, is obtained as in the one-level case by modifying the local A_j matrices into $\hat{A}_j$ matrices to express Robin interface conditions.

In the numerical experiments below, we consider the 2-D Laplace problem on the unit square, discretized using the 5-point finite difference stencil. Following [8], we obtain $\hat{A}_j$ using only a first-order accurate discretization of the normal derivative in the Robin conditions, which requires modifying only diagonal entries of A_j, namely those corresponding to the unknowns on the interfaces. As for the optimized value p^* of the parameter p, we follow again [8] and take, for the one- and two-level methods (i.e., ORAS and ORAS2)

$$p^*_{\text{one-level}} = 2^{-1/3}\pi^{2/3}(ovlp \cdot h)^{-1/3}, \tag{5}$$

$$p^*_{\text{two-level}} = 2^{-1/3}\pi^{2/3}(ovlp \cdot h)^{-1/3}(H_{x,y})^{-2/3}, \tag{6}$$

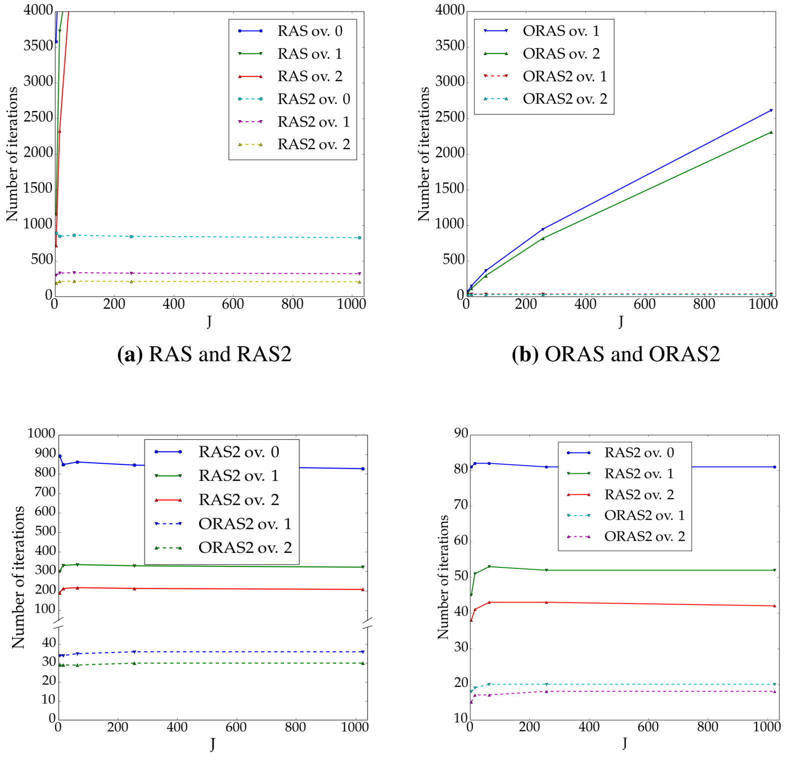
where h and H denote the fine and coarse mesh sizes. As for the value of the overlap $ovlp$, it has to be handled with some care: in the formulas (5)-(6), it is the geometrical (i.e., physical) overlap of the method, while the overlap value defined in PETSc is the number of extra mesh layers per subdomain at interfaces. An overlap of 1 in PETSc implies one extra mesh layer for both subdomains at an interface, thus an algebraic overlap of 2 (- an algebraic overlap of 0 corresponds to Block Jacobi). An algebraic overlap of 2 means a geometrical overlap of $3h$ for the RAS method and h for the (one- or two-level) ORAS method [10, 14], and thus $ovlp = 1$ in the above formulas. Similarly, an overlap of 2 in PETSc corresponds to an algebraic overlap of 4, that is a geometrical overlap of $5h$ for RAS and $3h$ for ORAS, and thus $ovlp = 3$.

To end this section, note that PETSc provides the `PCSetModifySubMatrices` tool to modify the diagonal values of the local matrices.

4 Weak Scalability Results

As stated earlier, we perform numerical experiments on the 2-D Laplace problem on the unit square discretized using the 5-point finite difference stencil. We perform a weak scalability analysis, that is, increase the size of the problem while maintaining constant the workload per processor. Each subdomain of the decomposition is handled by one CPU core (corresponding to one MPI rank). We increase the number J of subdomains/cores following the list $J = 4, 16, 64, 256, 1024, \ldots$ with decomposition into $J = \frac{1}{H} \times \frac{1}{H}$ subdomains on the unit square (H being the coarse mesh size as before). To maintain the workload per CPU core constant, the fine mesh size h is decreased proportionally, such that the ratio h/H remains constant as well as, in turn, the local problem size within each subdomain. We consider two workloads, the first one with a 256×256 fine mesh within each subdomain, the second (heavier) one with 512×512 local meshes, yielding a h/H ratio of .004 and .002, respectively.

Three different supercomputers were used to perform our tests: `Ada` and `Turing` at the Institute for Development and Resources in Intensive Scientific Computing (CNRS/IDRIS), and `Occigen` at the National Computing Center for Higher Education (CINES). The `Ada` and `Occigen` machines are meant for a wide-ranging usage and are composed of large memory SMP nodes interconnected by a high-speed InfiniBand network, for a cumulated peak performance of 233 Tflop/s and 3.5 Pflop/s, respectively. The `Turing` machine is an IBM Blue Gene/Q massively parallel architecture with a cumulated performance of 1.258 Pflop/s.



(a) RAS and RAS2

(b) ORAS and ORAS2

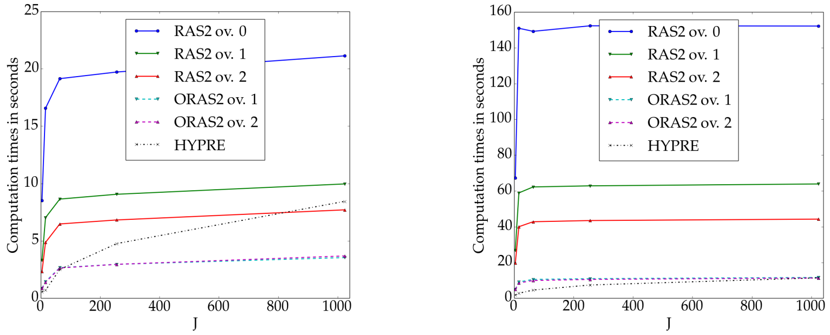
(c) RAS2 and ORAS2 (zoom)

(d) RAS2 and ORAS2 (GMRES zoom)

Fig. 2: Number of iterations in the weak scaling experiment with $h/H = .004$ (last panel with GMRES acceleration).

Fig. 2 shows in the first three pannels the number of stationary iterations obtained using the one- and two-level (O)RAS methods up to 1024 CPU cores. As can be seen on Figs. 2a and 2b, the one-level methods do not scale (here in terms of iterations), while the two-level methods do. Fig. 2c zooms on the two-level results of the previous plots, showing the superiority of the optimized methods. In Fig. 2d we show that using GMRES acceleration to the experiments in Fig. 2c lowers the iteration counts for all methods, but does not change the relative superior performance of the optimized methods compared to the classical ones. The equivalent zoomed plot obtained (with stationary iterations) using the classical approach consisting in choosing coarse grid nodes in the middle of each subdomain (circled points in Fig. 1) is visible on Fig. 5a. As expected, these results confirm the lower iteration count of the new approach already observed in [11].

Fig. 3 shows `Ada` timings for the two workloads ($h/H = .004$ and $h/H = .002$) with stationary iterations, using up to 1024 cores. As above in terms of iterations,

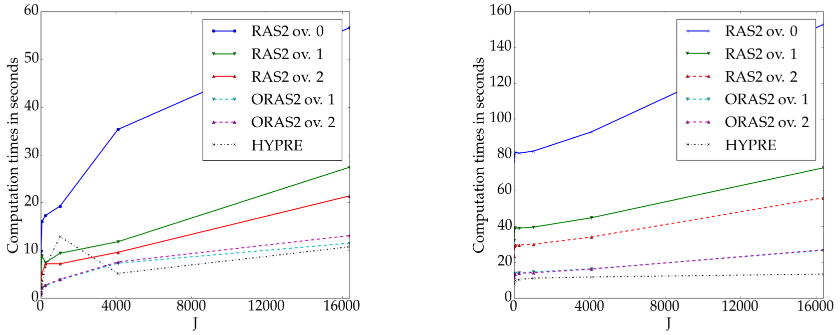


(a) $h/H = .004$ **(b)** $h/H = .002$

Fig. 3: Computation times (s.) for the weak scaling experiment up to 1024 cores on `Ada`, for the two different workloads. `HYPRE/BoomerAMG` is used with the default PETSc settings.

we here observe that the RAS2 and ORAS2 scale well in terms of computing time, with the optimized methods again showing superior performances. The use of a second layer of overlap does not appear beneficial in the ORAS2 method. On these plots also appear the corresponding results obtained using the multigrid library `HYPRE` as interfaced by PETSc, with the default settings. This amounts to using the `BoomerAMG` [13] component of `HYPRE`, for which the default settings are meant to work fairly well for two-dimensional diffusion problems [5]. The `HYPRE` results exhibit a scalability curve that is not as flat as the (O)RAS2 ones within this range of number of processors, with comparable computing times.

Numerical tests were pursued up to 16384 cores using the `Occigen` and `Turing` machines, as shown in Fig. 4. The scalability properties of the RAS2 and ORAS2

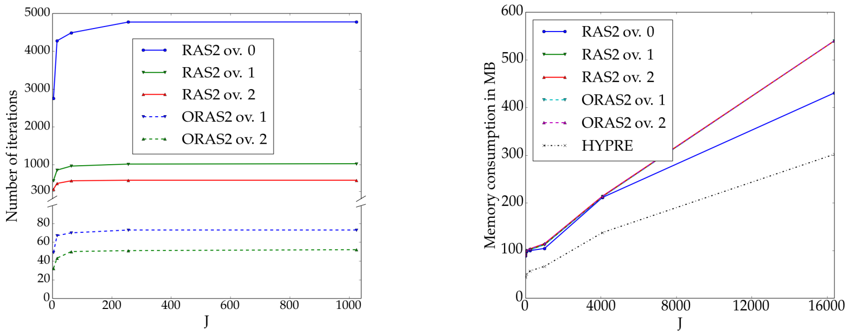


(a) `Occigen` timings **(b)** `Turing` timings

Fig. 4: Computation times (s.) for the weak scaling experiment up to 16384 cores on `Occigen` and `Turing` with $h/H = .004$.

methods remain decent, with the latter again performing better. As for the HYPRE results, they exhibit on Occigen (Fig. 4a) the expected scalability above 4092 cores, but not up to 1024 cores, as already observed above on Ada. This behavior remains unexplained to us, and has been observed repeatedly on these two machines of similar architecture. Changing architecture and running on Turing (Fig. 4b) however yields a flat scalability curve for HYPRE already below one thousand cores. The computing times on Turing are noticeably slower than on Occigen due to slower processors.

Finally, Fig. 5b shows the memory footprints of the different methods measured on Occigen. The overlapping RAS2 and ORAS2 methods yield very close footprints, which differ significantly from the non-overlapping ones only at 16384 cores, probably because of MPI scalability effects. Fig. 5b also shows that the HYPRE method yields the lowest memory footprint and it is unclear to us wether this comes from a better implementation or if it has a theoretical explanation.



(a) RAS2 and ORAS2 (zoom, coarse nodes in the middle of each subdomain).

(b) Average (on all the MPI tasks) of the maximal physical memory consumption.

Fig. 5: Number of iterations with the classical choice of coarse grid nodes (left) and memory footprint (right) in the weak scaling experiment with $h/H = .004$.

5 Conclusions

We implemented two improvements to the RAS method built in the PETSc library, namely a new coarse correction to obtain a (scalable) two-level method, as well as optimized interface conditions. This implementation was done using only existing PETSc tools, mainly preconditioner composition and submatrix modification.

We showed numerically that combining these two improvements yields substantial improvement on the standard RAS and, on a 2-D Laplace problem, the resulting ORAS2 method appears competitive with the multigrid HYPRE library up to 16k cores, despite a larger memory footprint.

Acknowledgements The authors wish to acknowledge Olivier Dubois from John Abbot Colllege (Quebec, Canada) for providing a first PETSc implementation of the methods presented here. The second author also wishes to acknowledge the help of his fellow CNRS/IDRIS team members. This work was performed using HPC resources from GENCI-CINES and GENCI-IDRIS.

References

1. Balay, S., Abhyankar, S., Adams, M.F., Brown, J., Brune, P., Buschelman, K., Dalcin, L., Dener, A., Eijkhout, V., Gropp, W.D., Kaushik, D., Knepley, M.G., May, D.A., McInnes, L.C., Mills, R.T., Munson, T., Rupp, K., Sanan, P., Smith, B.F., Zampini, S., Zhang, H.: PETSc users manual. Tech. Rep. ANL-95/11 - Revision 3.10, Argonne National Laboratory (2018). URL `http://www.mcs.anl.gov/petsc`
2. Balay, S., Abhyankar, S., Adams, M.F., Brown, J., Brune, P., Buschelman, K., Dalcin, L., Dener, A., Eijkhout, V., Gropp, W.D., Kaushik, D., Knepley, M.G., May, D.A., McInnes, L.C., Mills, R.T., Munson, T., Rupp, K., Sanan, P., Smith, B.F., Zampini, S., Zhang, H.: PETSc Web page. `http://www.mcs.anl.gov/petsc` (2018). URL `http://www.mcs.anl.gov/petsc`
3. Balay, S., Gropp, W.D., McInnes, L.C., Smith, B.F.: Efficient management of parallelism in object oriented numerical software libraries. In: E. Arge, A.M. Bruaset, H.P. Langtangen (eds.) Modern Software Tools in Scientific Computing, pp. 163–202. Birkhäuser Press (1997)
4. Cai, X.C., Sarkis, M.: A restricted additive Schwarz preconditioner for general sparse linear systems. SIAM J. Sci. Comp. **21**(2), 239–247 (1999)
5. Center for Applied Scientific Computing: Hypre user's manual. Tech. Rep. Software Version: 2.11.2, Lawrence Livermore National Laboratory (2017). URL `https://computation.llnl.gov/projects/hypre-scalable-linear-solvers-multigrid-methods`
6. Ciaramella, G., Gander, M.J.: Analysis of the parallel Schwarz method for growing chains of fixed-sized subdomains: Part I. SIAM Journal on Numerical Analysis **55**(3), 1330–1356 (2017)
7. Ciaramella, G., Gander, M.J.: Analysis of the parallel Schwarz method for growing chains of fixed-sized subdomains: Part II. SIAM Journal on Numerical Analysis **56**(3), 1498–1524 (2018)
8. Dubois, O., Gander, M.J., Loisel, S., St-Cyr, A., Szyld, D.B.: The optimized Schwarz methods with a coarse grid correction. SIAM J. Sci. Comp. **34**(1), A421–A458 (2012)
9. Efstathiou, E., Gander, M.J.: Why restricted additive Schwarz converges faster than additive Schwarz. BIT Numerical Mathematics **43**(5), 945–959 (2003)
10. Gander, M.J.: Schwarz methods over the course of time. ETNA **31**, 228–255 (2008)
11. Gander, M.J., Halpern, L., Santugini, K.: A new coarse grid correction for ras/as. In: Domain Decomposition Methods in Science and Engineering XXI, Lecture Notes in Computational Science and Engineering, pp. 275–284. Springer-Verlag (2014)
12. Gander, M.J., Loneland, A., Rahman, T.: Analysis of a new harmonically enriched multiscale coarse space for domain decomposition methods. arXiv preprint arXiv:1512.05285 (2015)
13. Henson, V.E., Yang, U.M.: Boomeramg: a parallel algebraic multigrid solver and preconditioner. Applied Numerical Mathematics **41**, 155–177 (2002)
14. St-Cyr, A., Gander, M.J., Thomas, S.J.: Optimized multiplicative, additive, and restricted additive Schwarz preconditioning. SIAM J. Sci. Comp. **29**(6), 2402–2425 (2007)