ParCFD'2020
32^{nd} International Conference on Parallel Computational Fluid Dynamics
May-11-13 2020, Nice FRANCE

AMG4PSBLAS LINEAR ALGEBRA PACKAGE BRINGS ALYA ONE STEP CLOSER TO EXASCALE

H. Owen*, G. Houzeaux*, F. Durastante◇, S. Fillipone† and P. D'Ambra°

* Barcelona Supercomputing Center (BSC),
Barcelona, Spain.
e-mail: herbert.owen@bsc.es, guillaume.houzeaux@bsc.es

◇Dipartimento di Matematica, Universit di Pisa, Pisa, Italy.
e-mail: f.durastante@na.iac.cnr.it

†Dept. of Civil Engineering and Computer Engineering, University of Rome Tor-Vergata,
Rome, Italy, and IAC-CNR.
e-mail: salvatore.filippone@uniroma2.it

°Institute for Applied Computing Mauro Picone (IAC), National Research Council (CNR),
Napoli, Italy.
e-mail: pasqua.dambra@cnr.it

Key words: AMG4PSBLAS, Multigrid, linear solver, real-world CFD, exascale

Abstract. In this work, we interfaced to the Alya code the development version of a software framework for efficient and reliable solution of the sparse linear systems for computation of the pressure field at each time step. We developed a software module in Alya's kernel to interface the current development version of the PSBLAS package (Parallel Sparse Basic Linear Algebra Subroutines) and the sibling package AMG4PSBLAS. PSBLAS implements parallel basic linear algebra operations and support routines for sparse matrix management tailored for iterative sparse linear solvers on parallel distributed-memory computers, supporting heterogeneity at the node level. It has gone under extension within the EoCoE-II project with the primary goal to face the exascale challenge. AMG4PSBLAS is a package of Algebraic MultiGrid (AMG) preconditioners built on the top of PSBLAS, which inherits all the flexibility and efficiency features of the PSBLAS infrastructure, and implements up-to-date AMG preconditioners exploiting aggregation of unknowns for the setup of the AMG hierarchy. Many preconditioners employing different aggregation schemes, AMG cycles, and parallel smoothers are available and were tested within the simulation carried out with the Alya code. Results show that the new solvers vastly outperform the original Deflated Conjugate Gradient method available in the Alya kernel in terms of scalability and parallel efficiency and represent a very promising software layer to move the Alya code towards exascale.

1 INTRODUCTION

Alya is an HPC-based multi-physics simulation code developed at BSC that has been designed to run efficiently in parallel supercomputers. Its target domain is engineering, with complex geometries and unstructured meshes. Based on a finite element discretization in space and both explicit and implicit time discretizations to solve compressible and incompressible flow problems, solids mechanics, and thermal coupling. Its scalability has been tested up to 100K cores. When implicit time discretization is used, the two main computational kernels are the creation of a matrix and right hand and the solution of the corresponding linear system. The second kernel is much more challenging for parallelization than the first one. As the problem's size increases, the communication between the different processors becomes more complicated and challenges scalability. In previous work, it has been shown Alya's scalability is not significantly compromised by this problem. However, as is well known, as the problem's size increases, a second problem of mathematical nature, challenges scalability: the number of iterations needed to converge to a specific tolerance increases when Krylov based solvers are used to solve the linear system. In Alya's own linear algebra package, this can be somehow alleviated but not overcome by deflation. Algebraic Multigrid can provide a much better weak scalability for unstructured grids but is much harder to code. In this work, we have preferred to interface Alya to the PSBLAS package (Parallel Sparse Basic Linear Algebra Subroutines) and of the sibling package AMG4PSBLAS [1].

2 INDUSTRIAL APLICATION

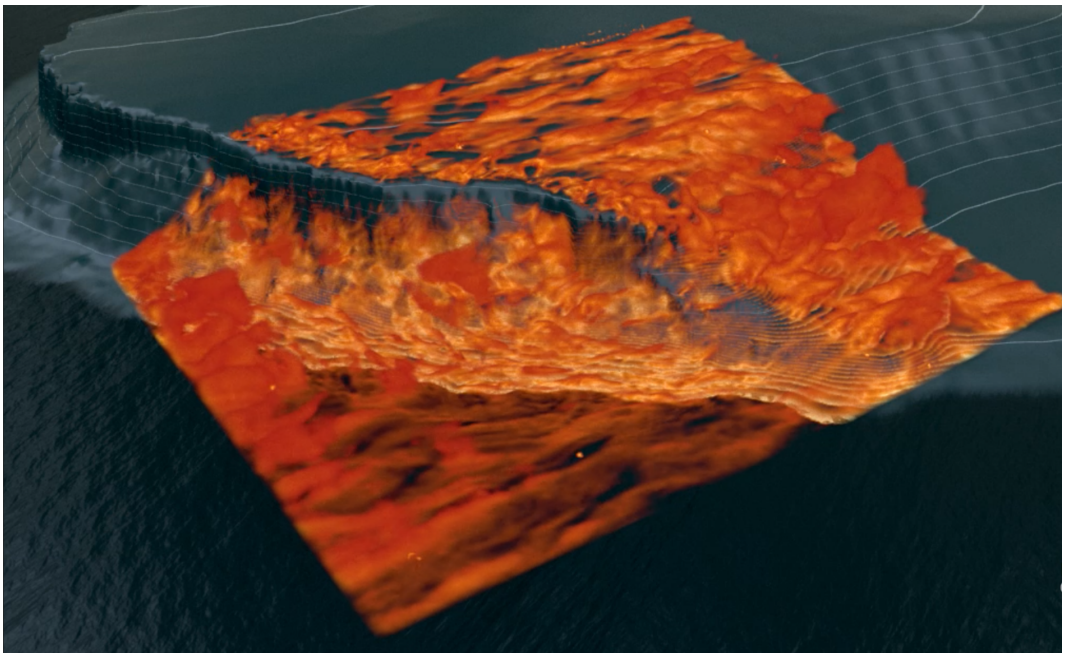

Figure 1: Flow over the Bolund cliff

Two real-world incompressible computational fluid dynamics problems are solved using the Vreman [2] Large Eddy Simulation turbulence model. The velocity is discretized explicitly using a 3rd order Runge Kutta scheme [3], and the pressure is treated implicitly. A low dissipation scheme based on pure Galerkin discretization and an EMAC (energy, momentum, and angular momentum conserving) scheme for the convective term lead to an efficient and accurate method free from numerical parameters [3]. A non-incremental fractional step method stabilizes the pressure and allows for equal order interpolation of velocity and pressure unknowns. The pressure field is obtained at each step by solving a discretization of a Poisson-type equation. PSBLAS package (Parallel Sparse Basic Linear Algebra Subroutines) and the sibling package AMG4PSBLAS are used to solve for the pressure with a Conjugate Gradient solver preconditioned with algebraic multigrid.

The first test case is based on the Bolund experiment, a classical benchmark for microscale atmospheric flow models over complex terrain [13]. Both strong and weak scalability tests are performed with an initial unstructured mesh of tetrahedra with 5.6M nodes. For weak scalability tests fixing different mesh sizes per cores, the mesh is uniformly refined [4] up to 358M nodes and tested up to 12288 cores. At each time step, we solve the spd linear systems arising from the pressure equation from an initial guess for pressure from the previous step and stop iterations when the Euclidean norm of the relative residual is not larger than TOL = 1e-3. A general row-block data distribution based on Metis 5.1 is applied for the parallel runs. The simulations have been performed with the Alya code interfaced to PSBLAS and AMG4PSBLAS, built with GNU compilers 7.2, on the Marenostrum 4 Supercomputer composed of 3456 nodes with 2 Intel Xeon Platinum 8160 chips with 24 cores per chip (ranked 30 in the Top 500 list [2], with more than ten petaflops of peak performance), operated by BSC. The facility was made available by a grant dedicated to the EoCoE II project from PRACE. Figure 1 shows a volumetric of the velocity field over the Bolund cliff.

The second test case is based on the NASA Common Research Model [5], a simplified geometry of a passenger aircraft. An initial mesh with 250M elements is used. Uniform mesh subdivision [4] is applied to reach 2000M elements to test weak scalability. Despite this second problem deals with external aerodynamics while the first one deals with Atmospheric Boundary Layer flow over complex terrain, the solutions strategy is nearly identical. Figure 2 shows streamlines of the flow around the wing.

To tests the solvers under realistic conditions, the cases are run on the coarse mesh until the flow is fully developed. The solution from the coarse mesh is used as an initial condition on the uniform subdivided mesh. The simulation is run further in the subdivided mesh until the solver's number of iterations stabilizes. 0.1 seconds are needed in the Bolund case. This velocity field is used as the initial condition for the solver convergence tests.

3 INITIAL RESULTS

To test the weak scalability of Alya with PSBLAS and AMG4PSBLAS, we have initially used the Bolund case. The initial mesh of 5.6M nodes was automatically uniformly refined

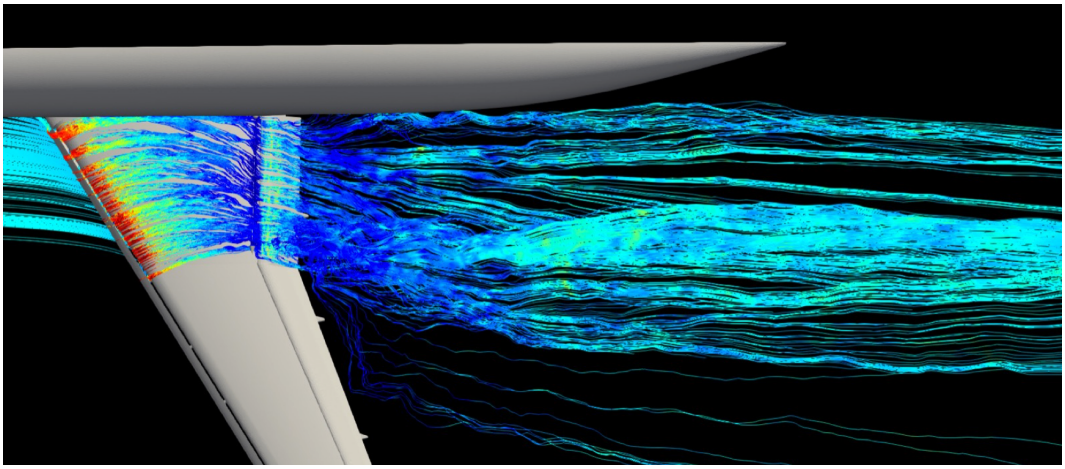

Figure 2: Flow around the NASA Common Research Model

Cores	Total Unk.	Av. Unk. per core	Mesh divisors	Iterations	CPU time [s]
48	5.6M	117k	0	5	0.67
96	5.6M	58k	0	5	0.33
192	5.6M	29k	0	5	0.17
384	44.8M	117k	1	8	1.02
768	44.8M	58k	1	8	0.53
1536	44.8M	29k	1	8	0.28
3072	358.4M	117k	2	4	0.72
6144	358.4M	58k	2	4	0.38
12288	358.4M	29k	2	4	0.25

Table 1: Number of iterations and CPU times

[4] to elements with one half the size leading to a mesh with eight times more nodes. A second refinement leads to elements with one quarter the size of the original mesh and 350M nodes. For each of the meshes, runs with three different numbers of elements were performed, leading to 9 nine runs. In Table 1, the number of iterations, CPU time, and key data for each run is summarized. For each of the meshes, the number of iterations is not altered by the number of cores used. One can see that the number of iterations increases when the first divisor is applied. However, it is later reduced with the second mesh divisor's application leading to excellent algorithmic scalability. This scalability can not be obtained with Krylov iterative solvers (not even with Deflation) available in Alya's linear Algebra package.

While looking at the number of iterations, we can verify that multigrid preconditioning has allowed Alya to overcome Krylov iterative solvers' limitations. The most critical parameter for Alya's users is the CPU time. For the CPU time to scale correctly, we need good mathematical behavior and satisfactory HPC implementation. When going from the original mesh to one with 64 times mores nodes, we observe a very good weak scaling

for the two simulations with a higher load per core. When the average number of nodes per core falls to 29k, the weak scalability suffers slightly but is still acceptable. When compared to Alya's implementation of the Deflated Conjugate Gradient, which takes 0.86 seconds on the finest mesh with 12288 cores, we see that PSBLAS/AMG4PSBLAS is 3.4 times faster. Moreover, on the finest mesh, PSBLAS/AMG4PSBLAS is faster with 3072 cores than Alya's solver with 12288 cores. From the trends we have observed, it is clear that as we move to larger problems in the path to exascale, the advantage of using multigrid will become even more noticeable.

4 CONCLUSIONS

Until recently, Alya has relied mainly on in-house developed code and minimized the usage of external libraries. The advantages are better control of Alya's evolution, ease of compilation, and guaranteed portability among different supercomputing platforms. However, minimizing the usage of external libraries has not allowed us to count with optimal linear algebra packages. Thanks to the collaboration established within the EoCoE II project, we are now starting to interface more with external libraries. While this does not mean that we need to abandon our developments, it gives us an alternative to overcome the limitations we faced in the past. In previous weak scalability studies with Alya [6] for a kiln furnace, it has been observed that convergence of the pressure degrades with the mesh size. Usage of the PSBLAS and AMG4PSBLAS has allowed Alya to obtain satisfactory weak scalability results for, including the pressure solution, the first time.

While the presented results are an essential step in the path of Alya towards exascale, the suitability of PSBLAS and AMG4PSBLAS for a wider range of problems needs to be tested. Moreover, porting to other supercomputers must be performed. Alya is one of the two CFD codes of the Unified European Applications Benchmark Suite (UEBAS) as well as the Accelerator benchmark suite of PRACE. It would be ideal to include Alya test cases that use PSBLAS and AMG4PSBLAS in such benchmarks. It is worthwhile mentioning that, for the moment, we have only tested the CPU implementation of PSBLAS and AMG4PSBLAS. Both libraries have already been ported to GPUs and tested in EU leading supercomputers such as Pizdaint. Finally, we would like to mention that interfacing with PSBLAS and AMG4PSBLAS has been far cheaper than trying to implement equivalent algebraic multigrid solvers inside Alya.

Acknowledgements

This work was supported by the Energy oriented Centre of Excellence II (EoCoE-II), grant agreement number 824158, funded within the Horizon2020 framework of the European Union.

REFERENCES

[1] Pasqua D'Ambra, Fabio Durastante, and Salvatore Filippone. Amg preconditioners for linear solvers towards extreme scale, 2021.

[2] A. W. Vreman. The filtering analog of the variational multiscale method in large-eddy simulation. *Physics of Fluids*, 15(8):L61–L64, 2003.

[3] O. Lehmkuhl, G. Houzeaux, H. Owen, G. Chrysokentis, and I. Rodriguez. A low-dissipation finite element scheme for scale resolving simulations of turbulent flows. *Journal of Computational Physics*, 390:51 – 65, 2019.

[4] Guillaume Houzeaux, Ral de la Cruz, Herbert Owen, and Mariano Vzquez. Parallel uniform mesh multiplication applied to a navierstokes solver. *Computers and Fluids*, 80:142 – 151, 2013. Selected contributions of the 23rd International Conference on Parallel Fluid Dynamics ParCFD2011.

[5] O. Lehmkuhl, G. I. Park, and P. Moin. Les of flow over the nasa common research model with near-wall modeling. In *Center for Turbulence Research,Proceedings of the Summer Program*, 2016.

[6] Mariano Vazquez, Guillaume Houzeaux, Seid Koric, Antoni Artigues, Jazmin Aguado-Sierra, Ruth Aris, Daniel Mira, Hadrien Calmet, Fernando Cucchietti, Herbert Owen, Ahmed Taha, Evan Dering Burness, Jose Maria Cela, and Mateo Valero. Alya: Multiphysics engineering simulation toward exascale. *Journal of Computational Science*, 14:15 – 27, 2016. The Route to Exascale: Novel Mathematical Methods, Scalable Algorithms and Computational Science Skills.