# Energetic Analysis of a Solver Stack

# for Frequency-Domain Electromagnetics

Emmanuel Agulloa, Luc Girauda, Stéphane Lanterib[[1]](#footnote-0) and Gilles Maraita

*aHiepacs project-team, Inria Bordeaux-Sud Ouest Research Center, Talence, France*

*bNachos project-team, Inria Sophia Antipolis-Méditerranée, Sophia Antipolis, France*

**Abstract**

We perform an energetic analysis of a software stack for the simulation of frequency-domain electromagnetic wave propagation. This solver stack combines a high order finite element discretization framework of the system of three-dimensional frequency-domain Maxwell equations with an algebraic hybrid iterative-direct sparse linear solver. This analysis is conducted on the KNL-based PCP system.

# Introduction

This work is concerned with the energetic analysis of the combined HORSE/MaPHyS numerical tool developed at Inria. The HORSE[[2]](#footnote-1) (High Order solver for Radar cross Section Evaluation) simulation software for implements an innovative high order finite element type method for solving the system of three-dimensional frequency-domain Maxwell equations. From the computational point of view, the central operation of a HORSE simulation is the solution of a large sparse and indefinite linear system of equations. High order approximation is particularly interesting for solving high frequency electromagnetic wave problems and, in that case, the size of this linear system can easily exceed several million unknowns. In this study, we adopt the MaPHyS[[3]](#footnote-2) hybrid iterative-direct sparse system solver, which is based on domain decomposition principles.

# Numerical approach

During the last 10 years, discontinuous Galerkin (DG) methods have been extensively considered for obtaining an approximate solution of Maxwell’s equations. Thanks to the discontinuity of the approximation, this kind of methods has many advantages, such as adaptivity to complex geometries through the use of unstructured possibly non-conforming meshes, easily obtained high order accuracy, *hp*-adaptivity and natural parallelism. However, despite these advantages, DG methods have one main drawback particularly sensitive for stationary problems: the number of globally coupled degrees of freedom (DoF) is much greater than the number of DoF required by conforming finite element methods for the same accuracy. Consequently, DG methods are expensive in terms of both CPU time and memory consumption, especially for time-harmonic problems. Hybridization of DG methods is devoted to address this issue while keeping all the advantages of DG methods. HDG methods introduce an additional *hybrid* variable on the faces of the elements, on which the definition of the local (element-wise) solutions is based. A so-called *conservativity condition* is imposed on the numerical trace, whose definition involved the hybrid variable, at the interface between neighbouring elements. As a result, HDG methods produce a linear system in terms of the DoF of the additional hybrid variable only. In this way, the number of globally coupled DoF is reduced. The local values of the electromagnetic fields can be obtained by solving local problems element-by-element. We have recently designed such a high order HDG method for the system of 3d time-harmonic Maxwell’s equations [1].

# Simulation software

HORSE is a computational electromagnetic simulation software for the evaluation of radar cross section of complex structures. This software aims at solving the full set of 3d time-harmonic Maxwell equations modeling the propagation of a high frequency electromagnetic wave in interaction with irregularly shaped structures and complex media. It relies on an arbitrary high order HDG method that is an extension of the method proposed in [1]. This HDG method designed on an unstructured possibly non-conforming tetrahedral mesh, leads to the formulation of an unstructured complex coefficients sparse linear system of equations for the DoF of the hybrid variable, while the DoF of the components of the electric and magnetic fields are computed element-wise from those of the hybrid variable. This software is written in Fortran 95. It is parallelized for distributed memory architectures using a classical SPMD strategy combining a partitioning of the underlying mesh with a message-passing programming model using the MPI standard. One important computational kernel of this software is the solution of a large sparse linear system of complex coefficients equations. In a preliminary version of this software, this system was solved using parallel sparse direct solvers such as MUMPS [2] or PaStiX [3]. However, sparse direct solvers are in general poorly scalable when it comes to solve very large linear system arising from the discretization of 3d problems. In this project, we study the possibility of improving the scalability of HORSE by considering the use of hybrid iterative/direct solvers whose design is based on domain decomposition principles.

# **MaPHyS algebraic solver**

The solution of large sparse linear systems is a critical operation for many numerical simulations. To cope with the hierarchical design of modern supercomputers, hybrid solvers based on algebraic domain decomposition methods have been proposed. Among them, approaches consisting of solving the problem on the interior of the domains with a sparse direct method and the problem on their interface with a preconditioned iterative method applied to the related Schur Complement have shown an attractive potential as they can combine the robustness of direct methods and the low memory footprint of iterative methods. MaPHyS (Massively Parallel Hybrid Solver) [4]-[5] is a parallel linear solver, which implements this idea. The underlying idea is to apply to general unstructured linear systems domain decomposition ideas developed for the solution of linear systems arising from PDEs. The interface problem, associated with the so-called Schur complement system, is solved using a block preconditioner with overlap between the blocks that is referred to as Algebraic Additive Schwarz. To cope with the possible lack of coarse grid mechanism that enables one to keep constant the number of iterations when the number of blocks is increased, the solver exploits two levels of parallelism (between the blocks using MPI and within the treatment of the blocks using threads). This allows exploiting a large number of cores with a moderate number of nodes, which ensures a reasonable convergence behavior. MaPHyS makes use of a sparse direct solver as a subdomain solver such as PaStiX (Parallel Sparse matriX package) or MUMPS. The parallelization of the direct solver relies on a specific partitioning of the matrix blocks; the core operations are multithreaded allowing a second level of parallelization. PaStiX and MUMPS make extensive use of highly optimized dense linear algebra kernels (e.g. BLAS kernels).

# Numerical and performance results

* 1. **Computing systems**

For the numerical simulations reported below we have used the **PRACE-PCP** Intel “Manycore” Knights Landing (KNL) cluster **Frioul**[[4]](#footnote-3) at CINES. Each node is a KNL 7250[[5]](#footnote-4) running at 1.4GHz with 68 cores and up to 4 threads per core (272 threads per node) with 16 GB of MCDRAM and 192 GB of RAM.

* 1. **MaPHyS used in standalone mode**

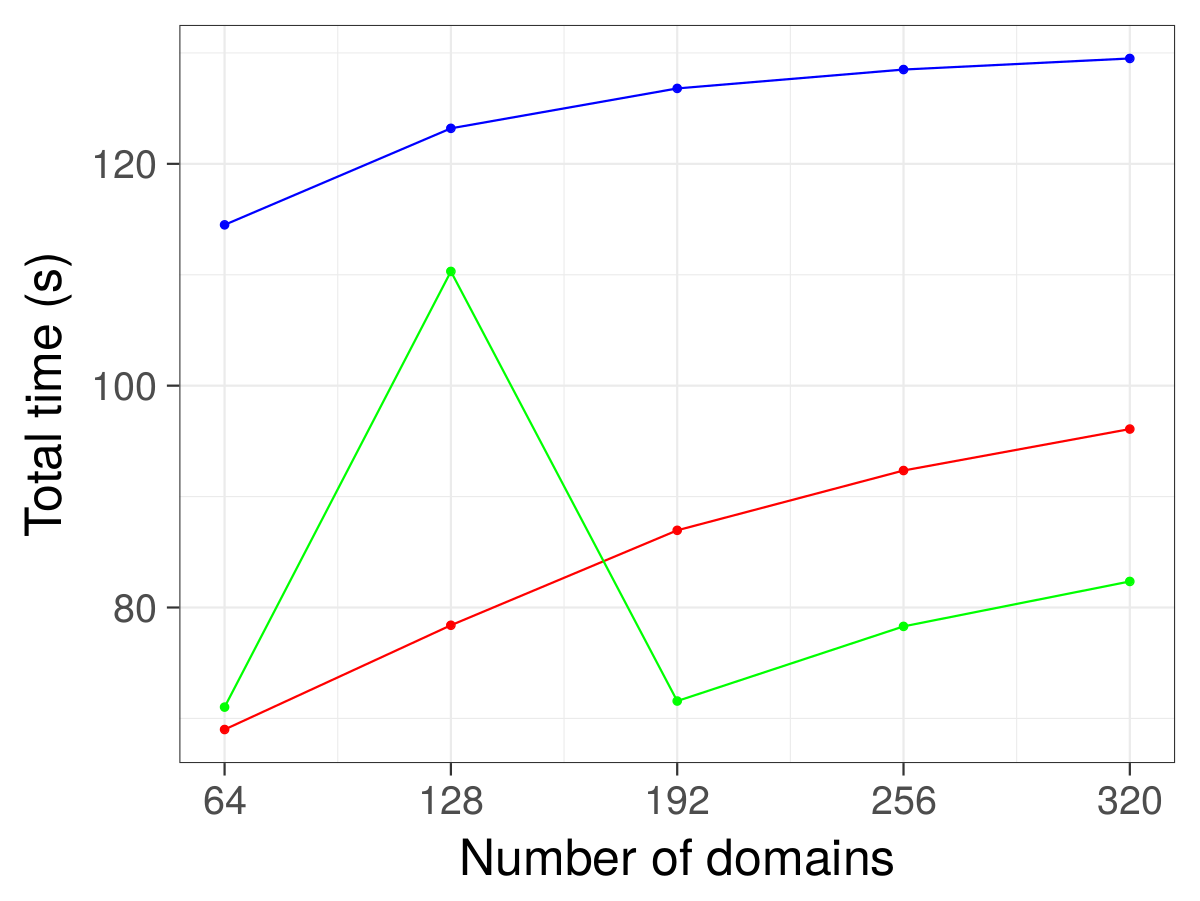
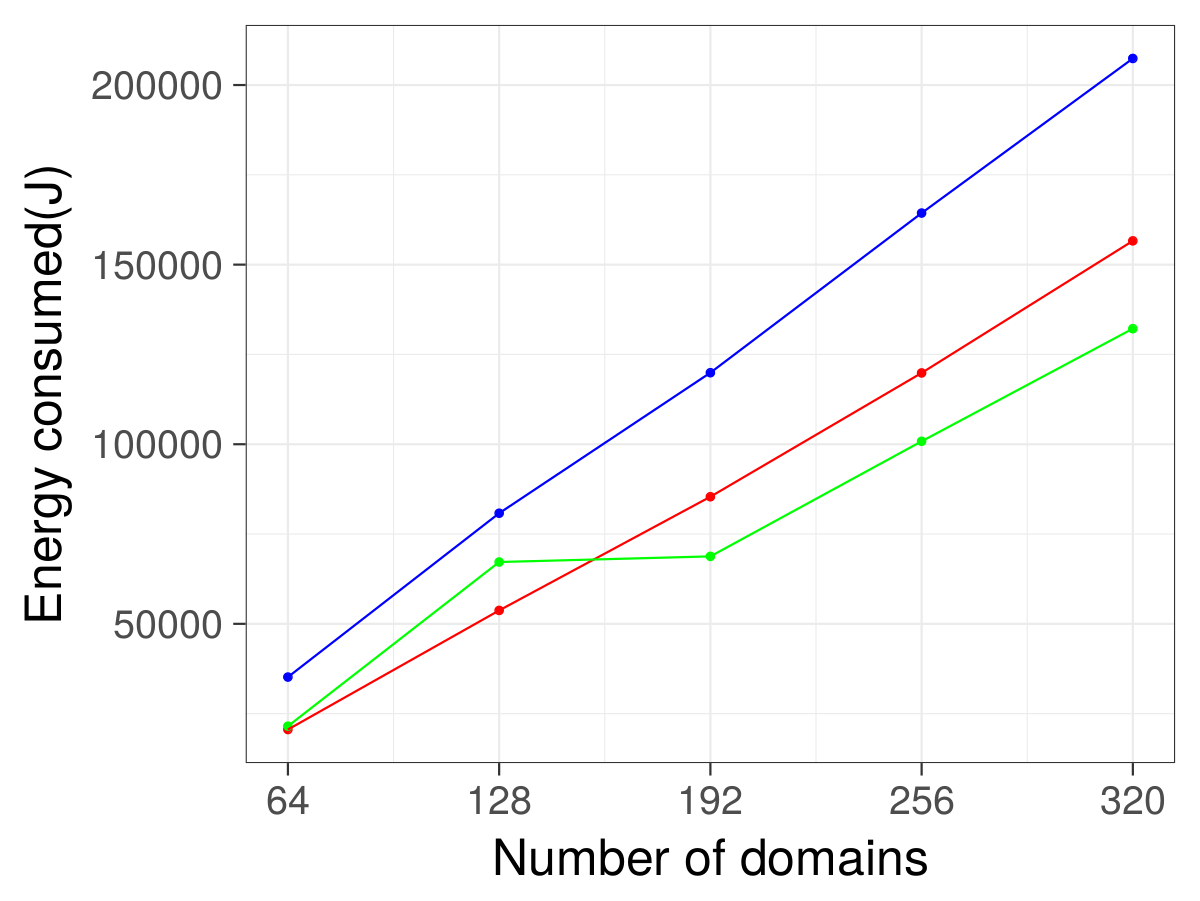
Weak scalability performance of the MaPHyS solver has been investigated in standalone mode. For these experiments we solve a 3D Poisson problem on a 2.5D domain that corresponds to a beam and a 1D decomposition. Each subdomain has at most two neighbors and is essentially a regular cube of size 403 (i.e., each subdomain has around 64,000 unknowns). The energy performance has been measured with Bull Energy Optimizer (BEO) as the total energy consumed by the job. We also had the opportunity to test Bull’s graphic tool Hdeeviz which shows detailed energy consumption over time (Figure 2). The additional metrics relevant for the performance of MaPHyS are the time for the factorization of interior subdomain unknowns, the time spent in the iterative solver, the number of iterations performed, and the total time spent in the solver. The local matrices are read from files, which is both time and energy consuming but not really relevant to MaPHyS performance since the matrices are usually computed locally and directly provided to the solver by the user.

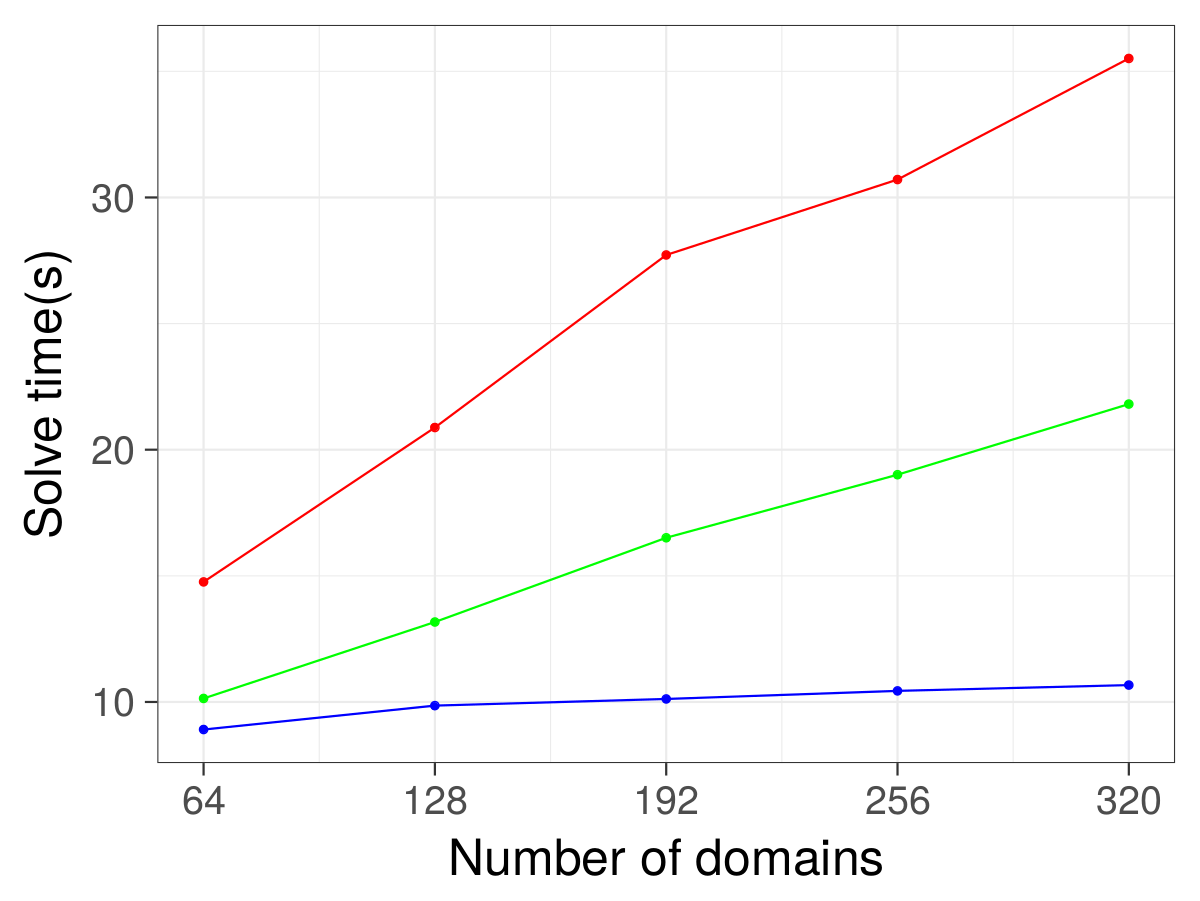
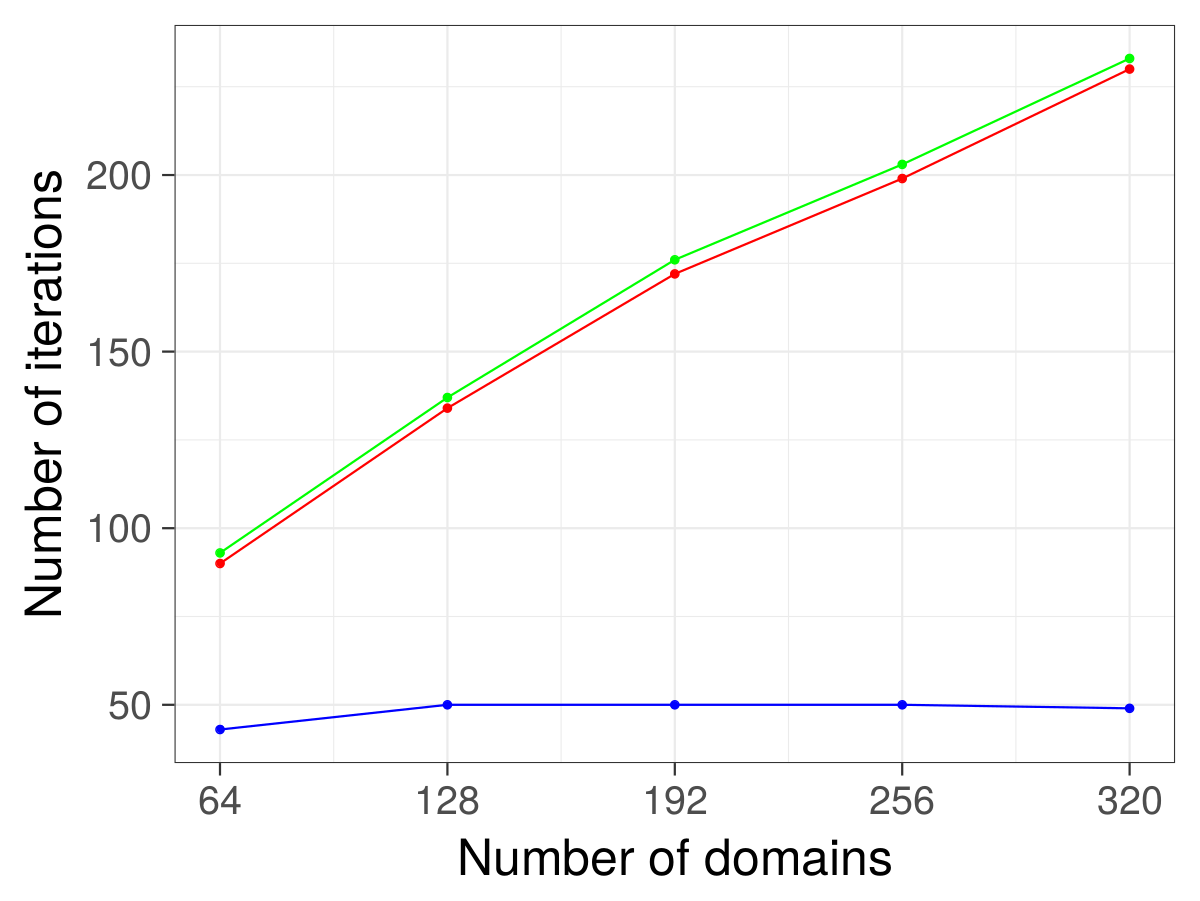
For our experiments, we consider three numerical configurations of the solver. In Figure 1, they are referred to as:

* dense: we consider the fully assembled local Schur complements to build the additive Schwarz preconditioner;
* sparse: the entries of the local dense Schur complements that are smaller than a given relative threshold (10-5) are discarded, the resulting sparse matrices are used to build the additive Schwarz preconditioner;
* dense+CGC: in addition to the previously described dense preconditioner a coarse grid correction [6] is applied to ensure that the convergence will be independant from the number of subdomains. In this experiment we compute five vectors per subdomain to create the coarse grid. The coarse grid being relatively small compared to the global problem, computations are centralized on one process and solved by the direct solver (MUMPS here).

Because the dense and sparse preconditioner do not implement any global coupling numerical mechanisms, the number of iterations is expected to grow as the number of subdomains for the 1D decomposition of the domain and our elliptic test example. This poor numerical behaviour can be observed in Figure 1-c, while it can be seen that the coarse space correction plays its role and ensures a number of iterations independent from the number of domains. This nice numerical behaviour translates in term of solution time for the iterative part where the variant with the coarse space correction outperforms the other two. However, the overhead of the setup phase for the construction of the coarse grid, which requires the solution of generalized eigenproblems, is very high and cannot be amortized if a single right-hand sides has to be solved (which is not the case for, e.g., radar cross section evaluation where many right-hand sides must be solved). The relative ranking of the variants with respect to the time to solution remains the same when we consider the energy criterion. However, the power requirements are different; using simple linear regression the power requirement for the dense preconditioner is around 5 kW, 8kW for the sparse and 10 kW for the two-level preconditioner. The high energy required by the two-level preconditioner is mainly due to the setup of the coarse space correction that is memory and cpu consuming. The fact that the sparse preconditioner is more demanding than the dense might be due to the more irregular memory pattern associated with it, that requires more memory traffic. As can be seen in Figure 2, the memory energy consumption represents a significant part of the total.

Figure 2 shows the detailed energy consumption over time for the case on one node with the dense preconditioner. One can see the setup and analysis parts of the run with low energy consumption. Then looking at the memory curve, one can identify the three steps of the MaPHyS solver. The iterative solver appears quite clearly as a large plateau where the energy cost is high for memory and low for CPU. It is consistent with the fact that this step is memory bound with many communications and relatively few computations. The total energy consumed by the node is 5.6 Wh = 20,160 J, which corresponds to the results given by BEO for this case.

 (a) Energy consumption (b) Total time in MaPHyS



(c) Number of iterations performed by MaPHyS (d) Time spent in the iterative part

Figure 1. Weak scaling of MaPHyS from 1 to 5 nodes, with 64 subdomains per node and 1 core per subdomain.



Figure 2. Energy consumption history for the dense preconditioner with hdeeviz (green=CPU, yellow=memory,cyan=total board).

|  |  |  |  |
| --- | --- | --- | --- |
| **Number of nodes** | **Number of domains** | **Global matrix size** | **Global Schur size** |
| 1 | 64 | 4,305,041 | 211,806 |
| 2 | 128 | 9,033,444 | 426,974 |
| 3 | 192 | 14,202,169 | 642,142 |
| 4 | 256 | 19,826,576 | 857,310 |
| 5 | 320 | 25,922,025 | 1,072,478 |

Table 1. Size of the global matrix and the global Schur complement matrix solved by MaPHyS in weak scaling.

* 1. **Scattering of a plane wave by a PEC sphere**

We now consider a more realistic problem that consists in the scattering of plane wave with frequency F=600 MHz by a perfectly electric conducting (PEC) sphere. the contour lines of the x-component of the electric field are visualized in Figure 3 left, and the obtained RCS is plotted in Figure 3 right together with a comparison with a reference RCS obtained from a BEM (Boundary Element Method) calculation. This problem is simulated using the coupled HORSE/MaPHyS numerical tool. The underlying tetrahedral mesh contains 37,198 vertices and 119,244 elements. We have realized a series of calculations for which the number of iterations of the MaPHyS interface solver has been fixed to 100. Simulations are performed using a flat MPI mode. We consider two mains situations: (a) the interpolation order in the HDG discretization method is uniform across the cells of the mesh; (b) the interpolation order is adapted locally to the size of the cell based on goal-oriented criterion. In the latter situation, we distribute the interpolation order such that there are at list 9 integration points (degrees of freedom of the Lagrange basis functions) per local wavelength. For the particular tetrahedral mesh used in this study,we obtain the following distribution of mesh elements: 12,920 (P1), 70,023 (P2), 31,943 (P3) and 4358 (P4). For a given mesh, a uniform interpolation order is not necessarily the best choice in terms of computational cost versus accuracy, especially if the mesh is unstructured as it is the case here. Increasing the interpolation order allows for a better accuracy at the expense of a larger sparse linear system to be solved by MaPHyS. By distributing the interpolation order according to the size of mesh cells allows for a good compromise between time to solution and accuracy. Performance and energy consumption figures are reported in Table 2. In this table, the number of subdomains also corresponds to the total number of core or MPI processes. The number of MPU processes per node can be deduced from the number of nodes. First of all, in most of the tested configurations, we observe a superlinear speedup, as a result of the reduction of the size of the local factors within each subdomain, which is not evolving linearly with the number of subdomains.

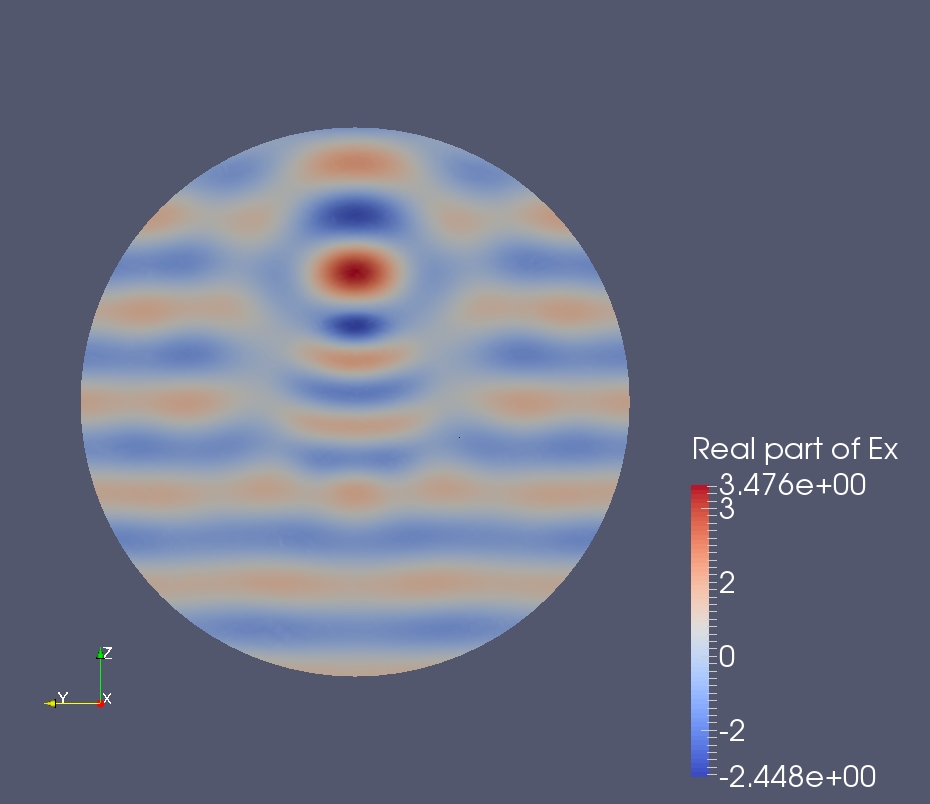
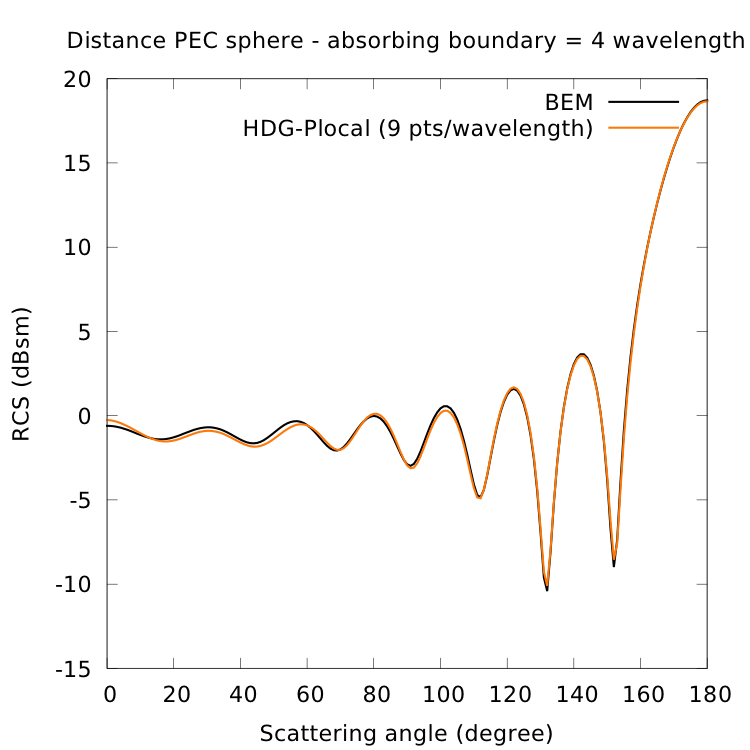
 

Figure 3. Scattering of a plane wave by a perfectly electric conducting sphere: contour lines of the x-component of the electric field (left) and RCS (right).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Method** | **Number of subdomains** | **Number of nodes** | **Wall time** | **Energy consumption** |
| **HDG-P1** | 16 | 1 | 143.0 sec |  |
|  | 32 | 2 | 52.4 sec |  |
|  | 64 | 4 | 21.0 sec |  |
|  | 64 | 8 | 20.2 sec |  |
|  | 128 | 16 | 9.5 sec |  |
| **HDG-P2** | 64 | 4 | 104.7 sec |  |
|  | 64 | 8 | 102.6 sec |  |
|  | 128 | 16 | 38.3 sec |  |
|  | 256 | 16 | 15.8 sec |  |
| **HDG-P3** | 64 | 8 | 415.7 sec |  |
|  | 128 | 16 | 130.5 sec |  |
|  | 256 | 16 | 48.7 sec |  |
| **HDG-P4** | 128 | 16 | 383.4 sec |  |
|  | 256 | 16 | 132.5 sec |  |
| **HDG-Pk, k=1,4** | 128 | 4 | 96.4 sec |  |
|  | 128 | 8 | 89.5 sec |  |
|  | 256 | 4 | 35.2 sec |  |
|  | 256 | 8 | 35.2 sec |  |
|  | 256 | 16 | 31.1 sec |  |

Table 2. Performance figures of the coupled HORSE/MaPHyS numerical tool. Scattering of a plane wave by a PEC sphere. Timings for 100 iterations of the interface solver of MaPHyS.

# Bibliographie

[1] L. Li, S. Lanteri and R. Perrussel. *A hybridizable discontinuous Galerkin method combined to a Schwarz algorithm for the solution of the 3d time-harmonic Maxwell’s equations.* J. Comp. Phys., Vol. 256, pp. 563-581 (2014)

[2] P.R. Amestoy, I.S. Duff and J.-Y. L'Excellent, *Multifrontal parallel distributed symmetric and unsymmetric solvers.* Comput. Methods in Appl. Mech. Eng., Vol. 184, pp. 501-520 (2000)

[3] P. Hénon, P. Ramet and J. Roman. *PaStiX: A high-performance parallel direct solver for sparse symmetric definite systems.* Paral. Comput.*,* Vol. 28, No. 2, pp. 301-321 (2002)

[4] L. Giraud, A. Haidar and L.T. Watson. *Parallel scalability study of hybrid preconditioners in three dimensions*. *Paral. Comput.*, Vol. 34, pp. 363-379 (2008)

[5] E. Agullo, L. Giraud, A. Guermouche and J. Roman. *Parallel hierarchical hybrid linear solvers for emerging computing platforms*. Compte Rendu de l'Académie des Sciences – Mécanique, Vol. 339, No. 2-3, pp. 96-105 (2011)

[6] E. Agullo, L. Giraud and L. Poirel, *Robust coarse spaces for Abstract Schwarz preconditioners via generalized eigenproblems*, Inria Research Report RR-8978 (2016)

**Acknowledgements**

This work was financially supported by the PRACE project funded in part by the EU’s Horizon 2020 research and innovation programme (2014-2020) under grant agreement 653838.

1. Corresponding author email address: stephane.lanteri@inria.fr [↑](#footnote-ref-0)
2. http://www-sop.inria.fr/nachos/index.php/Software/HORSE [↑](#footnote-ref-1)
3. https://gitlab.inria.fr/solverstack/maphys [↑](#footnote-ref-2)
4. https://www.cines.fr/le-supercalculateur-frioul [↑](#footnote-ref-3)
5. https://ark.intel.com/fr/products/94035/Intel-Xeon-Phi-Processor-7250-16GB-1\_40-GHz-68-core [↑](#footnote-ref-4)