Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 108C (2017) 818–827

Procedia Computer Science

International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland

Higher-Order Finite Element Electromagnetics Code for HPC environments

Daniel Garcia-Donoro[1], Adrian Amor-Martin[2], and Luis E. Garcia-Castillo[2]

[1] Xidian University, Xi'an, China.
daniel@xidian.edu.cn

[2] Department of Signal Theory and Communications, Leganes, Madrid, Spain.
[aamor,luise]@tsc.uc3m.es

Abstract

In this communication, an electromagnetic software developed to work in High-Performance Computing (HPC) environments is presented. The software is mainly based on the Finite Element Method (FEM) making use of a number of numerical techniques developed by its authors including its own family of higher-order curl-conforming elements and a non-standard mesh truncation methodology for the analysis of open region problems such as those of scattering and radiation. The code is written in FORTRAN 2003 and it adopts from scratch parallel programming paradigms such as Message Passing Interface (MPI) and Open Multi-Processing (OpenMP). It also includes an in-house development to ease the use of remote computer systems and HPC environments. Initially designed for single-user multicore machines and small cluster environments, a number of modifications have been included in the code in order to make it capable of running on large-scale computer systems and hence, be able to deal with larger problems in terms of number of unknowns. Details of the implementation are shown. Numerical results obtained on an HPC system corresponding to the analysis of a few illustrative challenging problems are also shown.

Peer-review under responsibility of the scientific committee of the International Conference on Computational Science

Keywords: Finite Element, Electromagnetics, HPC, MPI, OpenMP

1 Introduction

In this communication, a simulator that makes use of some of the research developments made within the research group the authors belong to is presented. In the last years a number of electromagnetic simulators have been introduced in the scientific community. The origin of these simulators is, usually, some legacy in-house codes which are constantly developing new features but keeping the same core. For this reason, the efficient use of HPC (high-performance computing) systems is difficult to achieve since many libraries have to be rewritten. Indeed, the simulator presented here, called HOFEM —higher order finite element method— is an in-house

1877-0509
Peer-review under responsibility of the scientific committee of the International Conference on Computational Science
10.1016/j.procs.2017.05.239

Table 1: Formulation magnitudes and parameters

	$\mathbf{V}$	$\bar{\bar{f}}_r$	$\bar{\bar{g}}_r$	h	$\mathbf{P}$	$\mathbf{L}$	Γ_{D}	Γ_{N}
Form. $\mathbf{E}$	$\mathbf{E}$	$\bar{\bar{\mu}}_r$	$\bar{\bar{\epsilon}}_r$	η	$\mathbf{J}$	$\mathbf{M}$	Γ_{PEC}	Γ_{PMC}
Form. $\mathbf{H}$	$\mathbf{H}$	$\bar{\bar{\epsilon}}_r$	$\bar{\bar{\mu}}_r$	$\frac{1}{\eta}$	$\mathbf{M}$	$-\mathbf{J}$	Γ_{PMC}	Γ_{PEC}

code that has been rewritten from scratch to adopt HPC paradigms such as Message Passing Interface (MPI, [1]) and Open Multi-Processing (OpenMP, [3]).

HOFEM kernel makes use of the Finite Element Method (FEM, see e.g., [12]) due to its versatility: e.g., complex geometries and non-homogeneous media can be characterized easily. In short, this method is based on the subdivision of the physical domain into simple geometrical shapes (tetrahedra, hexahedra, triangular prisms...) over which the solution is approximated by polynomials. Thus, the original partial differential equation can be translated into an algebraic system of equations. The correctness of the solution povided by the kernel has been already verified and validated, [9, 10].

Preliminary performance tests of the simulator were conducted on single-user multicore machines and small cluster environments. Recently, the code has experimented a number of modifications in order to make it capable of running on larger-scale computer systems and hence, be able to deal with larger problems in terms of number of unknowns. In this context, the motivation of this communication is to show the performance of HOFEM in such HPC environments.

The rest of the paper is organized as follows: the FEM formulation and the electromagnetic features of HOFEM are brielfly described in Section 2 , the computational features and flowchart of its execution in parallel are presented in Section 3, numerical results are presented in Section 4, and finally, some conclusions are provided in Section 5.

2 Electromagnetic Modeling Features

HOFEM makes use of a weak formulation based on the double curl vector wave equation to characterize the electromagnetic field in a given problem domain Ω^{FEM}. The formulation is applied in the frequency domain in terms of either electric ($\mathbf{E}$) or magnetic ($\mathbf{H}$) field

$$\boldsymbol{\nabla} \times \left(\bar{\bar{f}}_r^{\,-1} \boldsymbol{\nabla} \times \mathbf{V} \right) - k_0^2 \, \bar{\bar{g}}_r \, \mathbf{V} = -jk_0 h_0 \, \mathbf{P} - \boldsymbol{\nabla} \times \left(\bar{\bar{f}}_r^{\,-1} \mathbf{L} \right) \quad \text{in } \Omega^{\mathrm{FEM}} \tag{1}$$

where k_0 is the wavenumber in vacuum, $\mathbf{V}$ is the unknown field and $\mathbf{P}/\mathbf{L}$ are the impressed electric and/or magnetic currents within Ω^{FEM}. Table 1 shows the different magnitudes involved in the $\mathbf{E}$ and $\mathbf{H}$ formulations.

The boundary conditions considered are of Dirichlet, Neumann and Cauchy types:

$$\hat{\mathbf{n}} \times \mathbf{V} = \boldsymbol{\Psi}_{\mathrm{D}} \quad \text{over } \Gamma_{\mathrm{D}} \tag{2}$$

$$\hat{\mathbf{n}} \times \left(\bar{\bar{f}}_r^{\,-1} \boldsymbol{\nabla} \times \mathbf{V} \right) = \boldsymbol{\Psi}_{\mathrm{N}} \quad \text{over } \Gamma_{\mathrm{N}} \tag{3}$$

$$\hat{\mathbf{n}} \times \left(\bar{\bar{f}}_r^{\,-1} \boldsymbol{\nabla} \times \mathbf{V} \right) + \gamma \, \hat{\mathbf{n}} \times \hat{\mathbf{n}} \times \mathbf{V} = \boldsymbol{\Psi}_{\mathrm{C}} \quad \text{over } \Gamma_{\mathrm{C}} \tag{4}$$

where Γ_{D}, Γ_{N} and Γ_{C} stands for the boundaries where the Dirichlet, Neumann and Cauchy conditions, respectively, apply. Symbol $\hat{\mathbf{n}}$ is the outward unit vector to the considered boundary

surface. Above boundary conditions are typically used to model perfect electric and magnetic boundaries (PEC and PMC), waveports, and even exterior boundaries for open radiation and scattering problems. Symbol γ denotes the complex propagation constant of the corresponding waveport mode or exterior excitation wave.

Then, Galerkin Method is used on (1) to obtain the variational formulation of the problem as:

Find $\mathbf{V} \in \mathbf{H}(\text{curl})$ such that

$$c(\mathbf{F}, \mathbf{V}) = l(\mathbf{F}), \qquad \forall\, \mathbf{F} \in \mathbf{H}(\text{curl})_0 \tag{5}$$

where the bilinear and linear forms, $c(\mathbf{F}, \mathbf{V})$ and $l(\mathbf{F})$, are defined as follows

$$c(\mathbf{F}, \mathbf{V}) = \int_\Omega (\boldsymbol{\nabla} \times \mathbf{F}) \cdot \left(\bar{\bar{f}}_r^{\,-1} \boldsymbol{\nabla} \times \mathbf{V} \right) d\Omega - k_0^2 \int_\Omega (\mathbf{F} \cdot \bar{\bar{g}}_r \, \mathbf{V}) \, d\Omega \; + \; \gamma \int_{\Gamma_\text{C}} (\hat{\mathbf{n}} \times \mathbf{F}) \cdot (\hat{\mathbf{n}} \times \mathbf{V}) \, d\Gamma_\text{C} \tag{6}$$

$$l(\mathbf{F}) = -jk_0 h_0 \int_\Omega \mathbf{F} \cdot \mathbf{P} \, d\Omega \; - \; \int_{\Gamma_\text{N}} \mathbf{F} \cdot \boldsymbol{\Psi}_\text{N} \, d\Gamma_\text{N} - \int_{\Gamma_\text{C}} \mathbf{F} \cdot \boldsymbol{\Psi}_\text{C} \, d\Gamma_\text{C} \; - \; \int_\Omega \mathbf{F} \cdot \boldsymbol{\nabla} \times \left(\bar{\bar{f}}_r^{\,-1} \mathbf{L} \right) \, d\Omega \tag{7}$$

and function spaces involved are

$$\mathbf{H}(\text{curl})_0 = \{ \mathbf{W} \in \mathbf{H}(\text{curl}),\, \hat{\mathbf{n}} \times \mathbf{W} = 0 \;\text{ on }\; \Gamma_\text{D} \} \tag{8}$$

$$\mathbf{H}(\text{curl}) = \{ \mathbf{W} \in L^2,\, \boldsymbol{\nabla} \times \mathbf{W} \in L^2 \} \tag{9}$$

In addition to the above Dirichlet, Neumann, and Cauchy boundary conditions, periodic boundary conditions (PBC) are also considered to analyze antenna arrays using the infinite array approach. Usual electromagnetic excitations (exterior and interior) are supported such as general waveports where the excited field is calculated by solving the two-dimensional eigenvalue problem, lumped ports, impressed electric and magnetics currents and plane waves. Lumped RLC (resistance, coils and capacitors) elements are also supported in both serial and parallel configurations.

To perform the discretization of the computational domain, HOFEM makes use of its own second and third order isoparametric curl-conforming tetrahedral and triangular prismatic finite elements (e.g., see [5] and references therein). The curl-conforming elements used constitute a rigorous implementation of Nedelec's mixed order elements. Curl-conforming versions of the hexahedra and support of hybrid meshes combining different shapes are in progress. The above mentioned curl-conforming basis are interpolatory. However, the experience of some of the authors with *hp*-discretizations, [11], and the collaboration with Prof. Demkowicz of the Universty of Texas at Austin, [7], has allowed the development of a software module implementing hierarchical versions of basis functions that is under test at present. Thus, variable order discretizations and *p*-adaptive methodologies will be supported. Also, a preliminary version of *h*-adaptivity is under test. Combination of both approaches, h and p, will allow to work with $h + p$-adaptive approaches in the short term. The discretization process with above described finite element basis yields to an algebraic sparse system of equations which is solved using a direct solver as detailed later in Section 3.

For open region problems a non-standard mesh truncation technique called Finite Element - Iterative Integral Equation Evaluation (FE-IIEE, [8]) is implemented. Specifically, the integral equation representation of the exterior field to an auxiliary surface provides in an iterative fashion an improved version of the residual function $\boldsymbol{\Psi}_\text{C}$ of Cauchy boundary condition (7). Thus, an asymptotically exact absorbing boundary condition for FEM is implemented preserving the original sparse nature of the finite element matrices. Note also that the iterations only

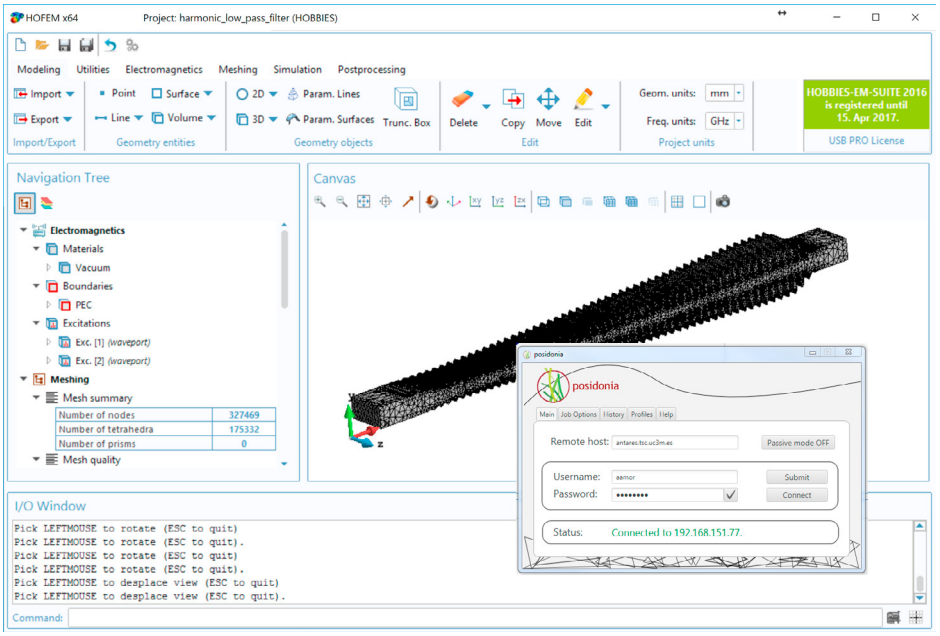


Figure 1: Snapshot of HOFEM GUI with posidonia attached as inset.

involve changes on the right hand side of the problem. That is, the FEM matrix needs to be factorized only once, and second and subsequent iterations only involve forward and backward substitutions when a direct solver is used.

3 Computational Features and Implementation

HOFEM is written using modern FORTRAN constructions, specifically FORTRAN 2003, with strong emphasis on code maintenance and reusability by means of the use of some object oriented programming (OOP) paradigms. In order to preserve the correctness of the code while changing or developing new modules, a complete set of automatic tests has been implemented. Some of these tests verify the electromagnetic kernel from the mathematical point of view using the Method of Manufactured Solutions [10]. The remaining tests execute electromagnetic problems with known solutions that can be easily checked. Thus, programming mistakes and errors may be detected when changing any module of the code or when a new one is included.

The input files needed by the electromagnetic kernel to perform the simulation may be generated through a graphical user interface (GUI) that provides an integral solution to the final user. To ease the use of HPC platforms, an in-house development of a general purpose HPC as a service (HPCaaS) interface has been developed, [6]. A snapshot of HOFEM GUI is shown in Figure 1 where the main page of this HPCaaS interface is displayed as inset.

To make possible an efficient use of HPC platforms, the code has been rewritten from scratch on the basis of the old sequential FEM code. At present, HOFEM implements a hybrid parallel methodology based on the use of MPI processes and OpenMP threads within each process. Thus, the code is able to exploit all the parallel capabilities available on HPC environments. Up to now, the code had been tested mainly on single-user multicore machines and small cluster

environments. Recently, the code has experimented a number of modifications in order to make it capable of running on larger-scale computer systems and hence, be able to deal with larger problems in terms of number of unknowns. To comment about the parallel implementation of HOFEM consider the flow chart shown in Figures 2 and 3. In this chart, a process is represented as a box (numbered as `p0`, `p1`, and so on) and a thread is plotted as a little box below the process specifying with color if the thread is used. Note that four threads per process are shown in the charts as an example. Obviously, the code can be run with a different number of threads per process.

The code starts calling the corresponding MPI subroutine that initializes the parallel environment and creates the process grid. This task is illustrated in the flow chart as MPI division. Once the process grid is created, each process reads the same input data which mainly consist of general parameters of the simulation and mesh information (node coordinates and element connectivities). The input data are expressed using LUA language constructions. It is important to mention that this read task works well when the number of MPI processes is less than approximately 10,000 processes. If the number of processes is higher, the file system could crash, so broadcast schemes must be considered. Once the input files are read, each process creates the same global *mesh object* and performs the global numbering of the degrees of freedom for each finite element. Each process executes the same code obtaining the same numeration, and in consequence, unnecessary communication is avoided. In order to save memory for later computations, each process creates now a local *mesh object* that contains only the information of the finite elements needed to compute the matrix coefficients in each process. Once the local *mesh object* is created in each process, the code checks if the FE-IIEE truncation method is enabled. In case the technique is enabled, the elements belonging to FE-IIEE boundaries (auxiliary and mesh truncation boundaries) are extracted from the global *mesh object*. Each process executes this extraction on its own obtaining the same information (again avoiding undesired communications). Once all the processes have a list with all the elements belonging to the mentioned boundaries, these elements are distributed uniformly to the processes and hence, the computational load of the method is splitted. After completing this task (or after checking if the FE-IIEE technique is not enabled), the global *mesh object* is deallocated releasing memory for future calculations.

At this point, HOFEM is ready to start the matrix filling process that calculates the matrix coefficients. Note that LHS is used in the charts to denote the left hand side of the matrix system of equations provided by the FEM discretization. RHS is used to denote the right hand side of the system which arises from the FEM discretization of the problem excitations. The matrix-filling task supports multi-thread execution for every process. Once all the coefficient are calculated, the factorization of the matrix is performed by calling one of the direct solvers supported by HOFEM (MUMP [2] or MKL Cluster PARDISO [4]). This task also supports multi-thread execution for every process.

Due to design constraints in the sparse solver used to solve the problem, the RHS coefficients are computed and stored only by process 0 (`p0`). The parallelization of this task is straightforward, however, as the RHS coefficients must be in `p0` at the beginning of the solving phase, we decided to compute everything directly in this process avoiding any type of communication. Even in the case of large problems, the computation time employed during this task is negligible compared to the simulation time. It is important to mention that HOFEM calculates the solution of the problem by block of excitations. Typically, ten excitations at a time are processed when solving problems with several dozens of millions of unknowns. This issue is due to the use of a centralized solution where all the solution coefficients are stored in `p0`. Future developments are already on progress to implement a distributed solution scheme to alleviate

this problem.

The last task on the simulation is to iteratively approximate the radiation condition by using the FE-IIEE method (if this technique is enabled). Firstly, HOFEM has to calculate the scattering field over the external boundary making use of an integral equation formulation. The calculations involved on this task are completely parallelized achieving high parallel performance as it will be shown in the next section. Each process only takes care of its boundary elements reducing the final scattering field at the end of calculation. Then, a new RHS is calculated and the system of equations is solved again (only back substitution is performed). If more than one frequency is set on the analysis, HOFEM repeats these steps from the matrix filling step to the end. The solution coefficients are stored on a binary file that is used to calculate different results such as nearfield, farfield, network parameters or power balance.

It is worth commenting that, due to memory issues when factorizing the matrix, the best partition scheme in this case has been to divide the full FEM matrix by rows. This memory issue has been identified (allocation of memory within the multifrontal solver proportional to the total bandwith of the global FEM matrix) and the authors are already working on a possible solution. However, up to now the only option to avoid it is reducing the bandwidth of the local matrices in each process to the minimum. The previous version of the code employed METIS to perform the partition between processes in terms of finite elements of the mesh. This leads to better quality local matrices (Schur complements of smaller size) than the current division by rows. However, the bandwidth of these local matrices is larger and the memory issue might also appear at some point.

4 Numerical experiments

As it was mentioned previously, the code has been already validated, and the objective here is to test the parallel performance of the code in an HPC enviroment. For that purpose, a number of challenging problems was analyzed. Numerical results of two of them are provided next. Both problems are open region problems in order to check the parallel performance of the whole code including the mesh truncation technique FE-IIEE. The HPC environment used is the cluster of Xidian University (XDHPC), which is equipped with 140 compute nodes connected by a 56 Gbps InfiniBand network. Each node has two twelve-core Intel Xeon 2690 V2 2.2 GHz CPUs with 64 GB of RAM and 1.8 TB of hard disk.

Firstly, a speedup benchmark has been carried out calculating the radar cross section (RCS) of a Chevrolet Impala car at 500 MHz using 120, 240, 360 and 480 cores respectively. The car is illuminated by eight plane waves coming from the cardinal directions N, S, E, W, NE, NW, SE and SW. The volumetric mesh has been generated using a uniform mesh size of 0.12λ obtaining 693,970 tetrahedra and a total number of unknowns equal to 4,518,248. The working frequency and mesh size have been chosen in order to be able to perform the simulation in 5 nodes of the cluster (120 cores) and to use more than 10% of the memory of each compute node when using 480 cores. The out-of-core version of the code is used. Figure 4 shows the speedup graph corresponding to the whole simulation considering factorization + solving + FE-IIEE truncation process. The parallel performance shown is close to 70%. Scalability is mainly limited by the use of direct solvers that typically exhibit a maximum figure which is around 60%. In the case of the memory used by HOFEM during this benchmark, the variation between 120 and 480 cores simulation differs on 28 GB. The total memory used by all processes during the simulation has been 135, 161, 157 and 153 GB, respectively. This data corresponds to the parallel configuration of 12 threads per MPI process.

The second numerical result presented here is the simulation of a long term evolution (LTE)

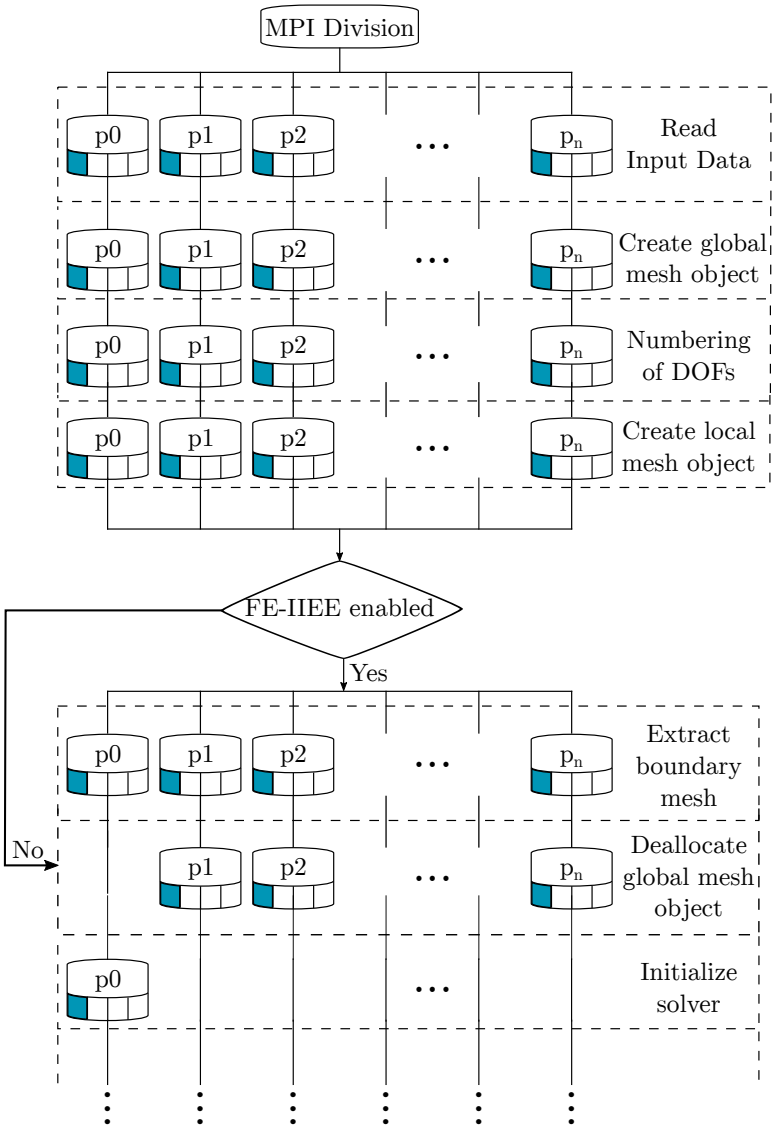


Figure 2: Flow chart for HOFEM, part 1.

4G base station antenna. This simulation is challenging due to the complexity of the feeding network for the whole array together with its electrical size. This base station antenna is composed of 64 elements with its corresponding feeding network. Geometrical model is shown in Fig. 5(b). The full wave analysis of the antenna is carried out between 2 and 3 GHz. The total number of excitations used in the simulation is 65. The length of the base station array is 1.6 meters, and the volumetric mesh used for each frequency contains 6,861,740 tetrahedra from which 45,121,862 unknowns are obtained. The simulation time of this structure is 330 minutes per frequency using 48 compute nodes and 1,152 CPU cores. The parallel configuration chosen in this case is 2 MPI processes on each node with 12 threads for each process. The out-of-core

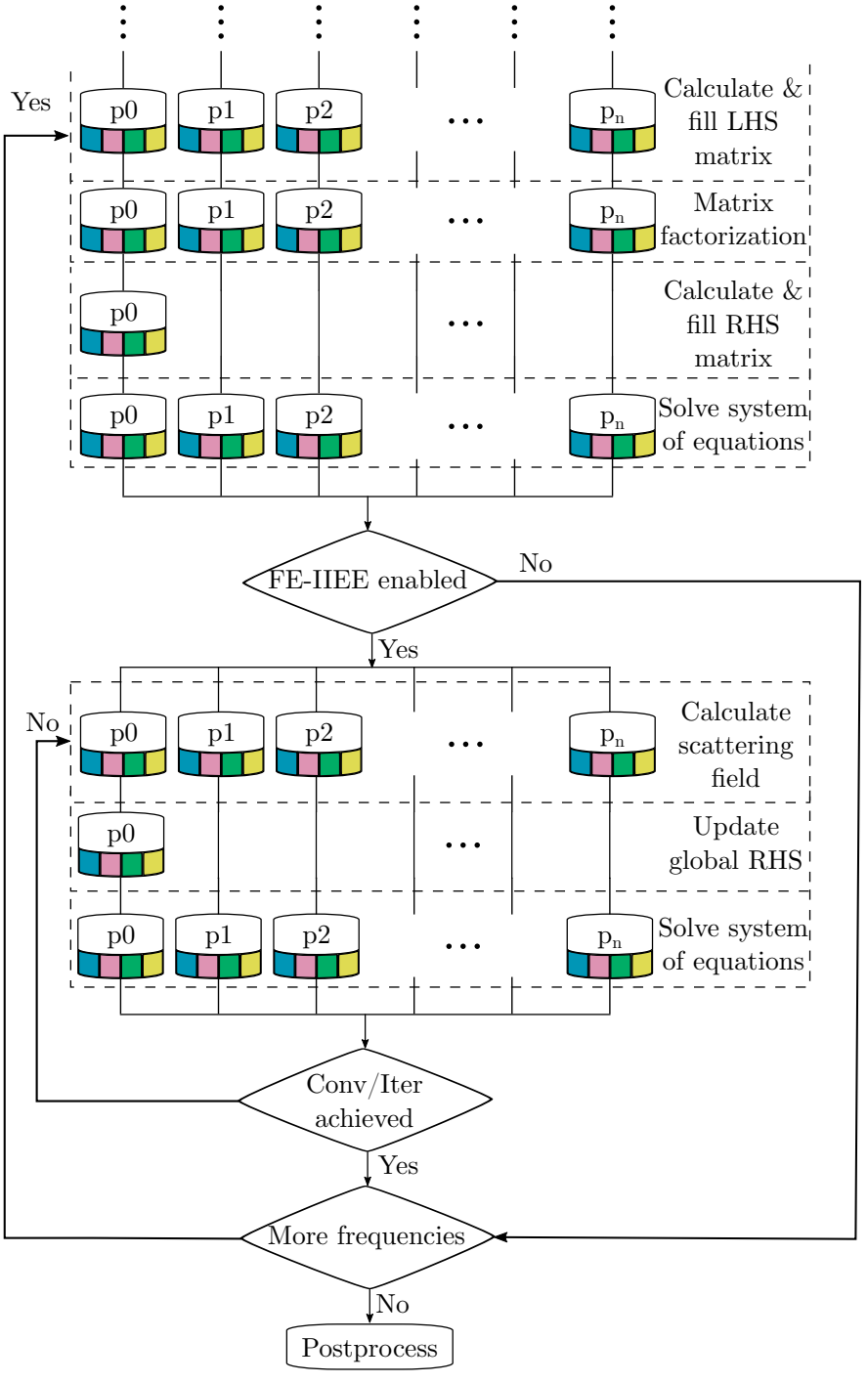


Figure 3: Flow chart for HOFEM, part 2.

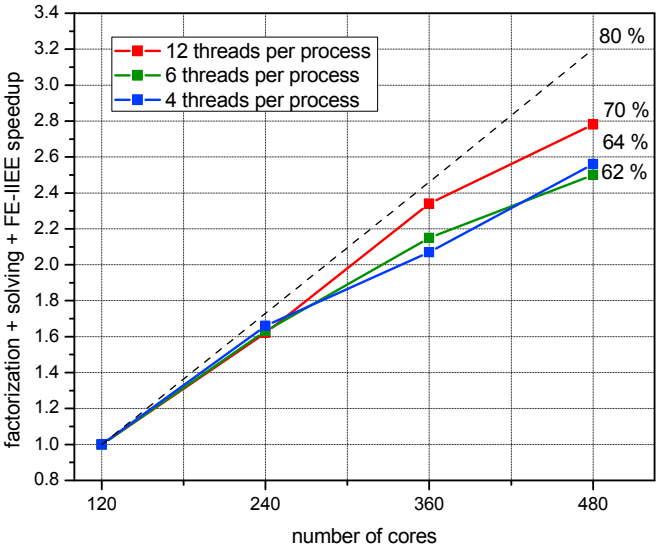
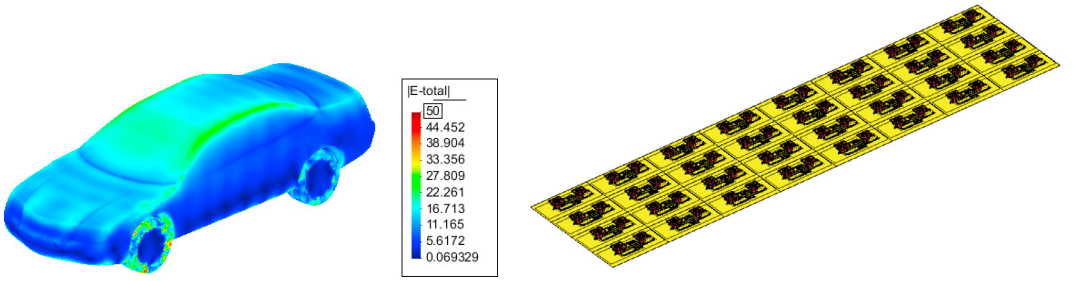


Figure 4: Speedup graph



(a) Magnitude of electric field (plane wave illuminates the front) on car

(b) Geometry of the LET array.

Figure 5: Some HOFEM simulations.

version of the solver is used employing 1.89 TB RAM.

5 Conclusions

In this communications an HPC EM simulator is presented. The code has demonstrated to be able to manage problems of several tens of millions of unknowns on more than one thousand cores with scalabilities close to 70%. Some memory issues have been identified and possible ways to overcome them have been pointed out.

5.1 Acknowledgments

This work has been financially supported by TEC2013-47753-C3, TEC2016-80386-P, CAM S2013/ICE-3004 projects and “Ayudas para contratos predoctorales de Formación del Profeso-

rado Universitario FPU".

References

[1] Message Passing Interface Forum. http://www.mpi-forum.org/.

[2] MUMPS Solver. http://www.enseeiht.fr/lima/apo/MUMPS/.

[3] OpenMP: The OpenMP API specification for parallel programming. http://openmp.org/.

[4] *Intel Math Kernel Library. Reference Manual*. Intel Corporation, 2009.

[5] Adrian Amor-Martin and L. E. García-Castillo. Second-order Nédélec curl-conforming prismatic element for computational electromagnetics. *IEEE Transactions on Antennas and Propagation*, 64(10):1–12, October 2016.

[6] Adrian Amor-Martin, Ignacio Martinez-Fernandez, and Luis E. Garcia-Castillo. Posidonia: A tool for HPC and remote scientific simulations. *IEEE Antennas and Propagation Magazine*, 6:166–177, December 2015.

[7] L. Demkowicz, Jason Kurtz, David Pardo, Maciek Paszynski, Waldemar Rachowicz, and Adam Zdunek. *Computing with hp Finite Elements. II Frontiers: Three Dimensional Elliptic and Maxwell Problems with Applications*. Chapman & Hall/CRC Press, Taylor and Francis, 2008.

[8] Raúl Fernández-Recio, Luis E. Garcia-Castillo, Sergio Llorente Romano, and Ignacio Gómez-Revuelto. Convergence study of a non-standard Schwarz domain decomposition method for finite element mesh truncation in electromagnetics. *Progress In Electromagnetics Research (PIER)*, 120:439–457, 2011.

[9] D. Garcia-Doñoro, I. Martinez-Fernandez, L. E. Garcia-Castillo, and M. Salazar-Palma. HOFEM: A higher order finite element method electromagnetic simulator. In *12th International Workshop on Finite Elements for Microwave Engineering*, Mount Qingcheng, Chendu, China, May 2014.

[10] Daniel Garcia-Doñoro, L. E. García-Castillo, and Sio Weng Ting. Verification process of finite-element method code for electromagnetics: Using the method of manufactured solutions. *IEEE Antennas and Propagation Magazine*, 7(2):28–38, April 2016.

[11] Ignacio Gomez-Revuelto, Luis E. Garcia-Castillo, and David Pardo. High-accuracy adaptive modeling of the energy distribution of a meniscus-shaped cell culture in a petri dish. *Journal of Computational Science*, 9:143–149, 2015. http://dx.doi.org/10.1016/j.jocs.2015.04.027.

[12] M. Salazar-Palma, T. K. Sarkar, L. E. García-Castillo, T. Roy, and A. R. Djordjevic. *Iterative and Self-Adaptive Finite-Elements in Electromagnetic Modeling*. Artech House Publishers, Inc., Norwood, MA, 1998.