Fast and Parallel Computational Techniques Applied to Numerical Modeling of RFX-mod Fusion Device

Domenico Abate [1,2], Bruno Carpentieri [3], Andrea G. Chiariello [4], Giuseppe Marchiori [2], Nicolò Marconato [2], Stefano Mastrostefano [1], Guglielmo Rubinacci [5], Salvatore Ventre [1], and Fabio Villone [1]

[1] DIEI, Università di Cassino e del Lazio Meridionale, Loc. Folcara, 03043 Cassino (FR), Italy
s.mastrostefano@unicas.it, ventre@unicas.it, villone@unicas.it

[2] Consorzio RFX, Corso Stati Uniti 4, Padova, Italy
domenico.abate@igi.cnr.it, giuseppe.marchiori@igi.cnr.it, nicolo.marconato@igi.cnr.it

[3] Nottingham Trent University, School of Science and Technology, Burton Street, Nottingham NG1 4BU, UK
bruno.carpentieri@ntu.ac.uk

[4] DIII, Seconda Università di Napoli, Via Roma 29, Aversa (CE), Italy
andreagaetano.chiariello@unina2.it

[5] DIETI, Università di Napoli Federico II, Via Claudio 21, 80125, Napoli
rubinacci@unina.it

Abstract ― This paper presents fast computational techniques applied to modelling the RFX-mod fusion device. An integral equation model is derived for the current distribution on the active coils of the conducting structures, and the input-output transfer functions are computed. Speed-up factors of about 200 can be obtained on hybrid CPU-GPU parallelization against uniprocessor computation.

Index Terms ― Fusion plasma devices, GPUs, HPC, integral formulation, parallelism.

I. INTRODUCTION

Modelling fusion devices is computationally very challenging due to the electromagnetic interaction of the fusion plasma and the surrounding conducting structures, which makes the problem inherently multiphysics. The evolution of the plasma may exhibit unstable modes, thus exacerbating the aforementioned problems and requiring a feedback controller. The design of such control system requires rather accurate response model of the overall system plasma plus conductors. Therefore, fast parallel techniques are often required to make the computations affordable [1, 2]. In this paper, we analyze the RFX-mod device [3], a medium size (major radius R = 2 m, minor radius a = 0.46 m) toroidal device particularly suited to explore innovative concepts in plasma control. Passive and active conductors are very important to determine the overall properties and performances of such feedback system and therefore they should also be adequately represented in any realistic model. The main conducting structures are the vessel (needed to have the vacuum inside the machine), the shells (highly conducting sheets needed for passive stabilization), the mechanical structure, hosting the active control coils. Figure 1 shows the 3D hexahedral mesh used.

In particular, RFX-mod is equipped with a state-of-the-art control system made by 192 (4 poloidal x 48 toroidal) independently fed active coils (Fig. 1), with more than 600 magnetic sensors acquired in real time. This makes RFX-mod on the one hand very challenging for numerical modelling but on the other hand an ideal test-bed for validating the predicting capabilities of computational tools. We compute the input-output transfer functions of the system, assuming as input the currents or the voltages of the active coils and as output suitable magnetic measures [4]. The presence of an axisymmetric plasma evolving through equilibrium states is self-consistently taken into account [1].

The computer solution of such a problem is very expensive, due to the complexity of the 3D geometry and the plasma contribution. The use of High Performance Computing (HPC) cluster is mandatory. The GPU architecture has a large amount of cores designed to run a large number of execution threads at the same time; the computational model used is the single instruction,

Submitted On: December 31, 2015
Accepted On: May 16, 2016

multiple data (SIMD), where concurrent threads execute the same code (called Kernel) on different data. In the present work, we focus our attention on a hybrid multi-node system for modeling RFX-mod devices.

The paper is organized as follows. Section II describes the model, while in Section III we illustrate the computational technique. Section IV reports the results and draws the conclusions.

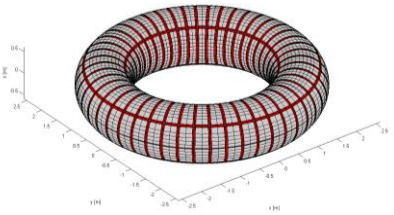

Fig. 1. Mesh used for the analysis of the problem.

II. MODEL

We consider a system of 3D conductors V_c discretized with a finite elements mesh. We use an integral formulation, which assumes as primary unknown the current density in V_c. We introduce the electric vector potential **T**, such that $\mathbf{J} = \nabla \times \mathbf{T}$, and then we expanded **T** in terms of edge elements $\mathbf{N}_k$, we have:

$$\mathbf{J} = \sum_k I_k \nabla \times \mathbf{N}_k \,. \tag{1}$$

Imposing Ohm's law in weak form, we get [1,2,8]:

$$\underline{\underline{L}} \frac{d\underline{I}}{dt} + \underline{\underline{R}}\,\underline{I} + \frac{d\underline{U}}{dt} = \underline{V}, \tag{2}$$

$$L_{i,j} = \frac{\mu_0}{4\pi} \int_{V_c} \int_{V_c} \frac{\nabla \times \mathbf{N}_i(\mathbf{r}) \cdot \nabla \times \mathbf{N}_j(\mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|} dV\, dV', \tag{3}$$

$$R_{i,j} = \int_{V_c} \nabla \times \mathbf{N}_i \cdot \boldsymbol{\eta} \cdot \nabla \times \mathbf{N}_j dV \,. \tag{4}$$

In these equations, $\underline{I}$ is the vector of degrees of freedom I_k in (1), $\underline{V}$ is the vector of externally applied voltages and $\boldsymbol{\eta}$ is the resistivity tensor. Matrix $\underline{\underline{L}}$ is a fully populated square matrix, which is the 3D analogue of mutual inductance of a system of magnetically coupled conductors; conversely, $\underline{\underline{R}}$ matrix is sparse and represents the resistance matrix of the 3D conductors. The quantity $\underline{U}$ is the magnetic flux due to plasma currents [1, 8]:

$$\underline{U} = \underline{\underline{M}}\,\underline{j}_S\,,\ \underline{j}_S = \underline{\underline{S}}\,\underline{\hat{\psi}}_e\,,\ \underline{\hat{\psi}}_e = \underline{\underline{Q}}\,\underline{I}\,, \tag{5}$$

where $\underline{j}_S$ are equivalent currents located on a coupling surface, $\underline{\underline{M}}$ is a mutual inductance matrix between the equivalent current and the 3D conducting structures, $\underline{\hat{\psi}}_e$ is the external magnetic flux, $\underline{\underline{Q}}$ is a matrix representing Biot-Savart integral [1] and $\underline{\underline{S}}$ is the plasma response matrix [8].

Combining (2)-(5), finally we get [8]:

$$\underline{\underline{L}}^* \frac{d\underline{I}}{dt} + \underline{\underline{R}}\,\underline{I} = \underline{V}\,, \quad \underline{\underline{L}}^* = \underline{\underline{L}} + \underline{\underline{M}}\,\underline{\underline{S}}\,\underline{\underline{Q}}, \tag{6}$$

to which we can add the expression for the magnetic field and flux perturbations $\underline{y}$ at given points, linearly related to 3D currents through a suitable matrix $\underline{\underline{C}}$ [1,8]:

$$\underline{y} = \underline{\underline{C}}\,\underline{I}\,. \tag{7}$$

Equations (6)-(7) represent the model; they can be easily recast in standard state space form. In the present paper, they are used to get the frequency-domain transfer functions between the inputs (voltages or currents in active coils) and the outputs (linear combinations of magnetic measurements). In doing so, the inversion of a complex matrix is required. Indeed, we split the unknowns into three subsets; the corresponding subset of indices of the various matrices are identified with the following suffix: "p" (passive structures), "m" (measurement coils), "a" (active coils), so that Equation (6) reads as:

$$\begin{aligned} &\left(j\omega \underline{\underline{L}}^*_{pp} + \underline{\underline{R}}_{pp} \right) \underline{I}_p + j\omega \underline{\underline{L}}^*_{pa} \underline{I}_a = 0, \\ &\underline{\underline{L}}^*_{mp} \underline{I}_p + \underline{\underline{L}}^*_{ma} \underline{I}_a = \underline{\Phi}_m, \end{aligned} \tag{8}$$

where $\underline{\Phi}_m$ represent the fluxes induced at measurements coils (i.e., the output of the system). After some simple algebraic manipulations it turns out:

$$\begin{aligned} &\underline{I}_p = -j\omega \left(j\omega \underline{\underline{L}}^*_{pp} + \underline{\underline{R}}_{pp} \right)^{-1} \underline{\underline{L}}^*_{pa} \underline{I}_a = \underline{\underline{H}}(j\omega) \underline{I}_a, \\ &\underline{\Phi}_m = (\underline{\underline{L}}^*_{mp} \underline{H}(j\omega) + \underline{\underline{L}}^*_{ma}) \underline{I}_a = \underline{\underline{T}}(j\omega) \underline{I}_a. \end{aligned} \tag{9}$$

Equation (9) can be used to evaluate Φ_m for a given assigned unitary current flowing in excitation coil. This transfer function can be used to design the feedback control. For each frequency and each active coil, we set Ia to 1 (the known terms in the Eq. (9)) and find the Flux for all measurement coils (unknowns variables).

III. FAST TECHNIQUES

In order to speed up the overall computation we move in two directions:

- parallelize the matrix assembly phase;
- accelerate the inversion of system (9).

A. Parallel assembly strategies

Matrices $\underline{\underline{L}}$ and $\underline{\underline{Q}}$ are very expensive to assemble. For $\underline{\underline{L}}$ matrix, parallelization can be achieved grouping elements of nodes into boxes, distributing boxes among processors, and performing the element-element integration independently on each processor. The locally assembled matrix is then compressed (see [2]).

The computation of matrix $\underline{\underline{Q}}$ is the most time consuming part of the assembly algorithm. In order to reduce this cost, in the present work, parallel assembly is implemented on multi CPUs and multi GPUs environment. Here we take advantage of the fact that

Biot-Savart integral computation for elements and field points, are independent from each other. Of course, different implementations are necessary to adapt the parallel computation to the two different hardware architectures. In the following, we briefly recall the main features of the two algorithms.

In multi CPUs environment we propose a standard parallel strategy using a simple domain decomposition approach that distributes the field points equally among the processors. After the local computations, a reduction operation is required to retrieve the complete matrix from each MPI process. This strategy scales linearly with the number of the processors.

In multi GPUs environment we propose to assign to each computational thread the evaluation of a contribution of the Biot-Savart integral corresponding to a given element and a given field point. All the contributions are summed on the CPU. The algorithm is briefly summarized in the follow, see [1] for details:

1) Allocate temporary data for storing the local contribute (CPU).
2) Compute the considered element and source point from the thread and block index (GPU).
3) Compute the shape function related to the considered element (GPU).
4) Compute the local contribution (GPU).
5) Return the partial matrix to the host memory (CPU).
6) Scatter the output data on the complete matrix on the host (CPU).

The final step is due to uncoalescent memory access needed to store the results in the final matrix and possible race conditions when two different contributions are summed in the same location. The dimension of the matrix can be huge compared to the on board GPU memory (which is typically of a few GB). Step 5 involves memory transfer from GPU to Host memory, but fortunately this has no impact on the overall performance of the code. We point to [9] for more sophisticated approaches not considered here.

B. Speed up of the linear system inversion

As far as the inversion of the linear system involved in (9) is concerned, it is worth noting that $\underline{\underline{L}}^{*}_{xy}$ are fully populated submatrices of matrix $\underline{\underline{L}}^{*}$ and $\underline{\underline{R}}_{pp}$ is a sparse positive definite matrix. Using a direct solver, the cost of the inversion procedure is $O(N^3)$, N being the number of unknowns present in the passive part of the device. When geometric details are added and/or a great accuracy is required in the computation, it is easy to exceed quickly the computational resources available on a uniprocessor system. The use of powerful computing facilities can help in the search of additional speed and increase the size of the solvable problems [5].

Nevertheless, there are cases in which parallelization fails poorly. For this problem, an approximated compression technique is mandatory. The authors successfully applied these methods for the study of plasma fusion devices [2] as well as in other fields (e.g., NDT [6]). These techniques are based on an effective low-rank approximation of the submatrix representing the far interaction between well separated parts of the device. The matrix-by-vector product $\underline{\underline{L}}^{*}_{ij} I_j$ related to these parts is replaced by an accurate low cost operator (the complexity is asymptotically only $O(N)$). Finally, the inversion in (9) can be performed by an iterative method (such as the GMRES method). It is worth noting that the preconditioner (essential for any iterative solver) is $\underline{\underline{R}}^{-1}_{pp}$, which can be computed in fast and accurate way by the means of Cholesky decomposition. It is important to stress that its factorization and back substitution is very cheap using a single CPU. Moreover the preconditioner turns out to be very effective, being the number of iterations required to converge very small.

IV. RESULTS

The computational cluster used for the evaluation of the numerical performances is made by two nodes. Each node consist of 16x cores Intel Xeon CPU E5-2690 (@ 2.90 GHz processor, 20 MB L2), 128 GB RAM, 2×NVIDIA Kepler K20 (2496 cores, 6 GB VRAM).

A. 2D validation and transfer function computation

First of all, a numerical validation of the procedure is carried out. We generated a 3D mesh which fictitiously reproduce an axisymmetric geometry, so that a 2D code (CREATE_L [7]) can be used as benchmark. We computed the transfer function $\underline{\underline{T}}$ defined in the previous section with the two codes, finding a very good agreement, as shown in Fig. 2. This confirms the correctness of the procedure.

In order to show the actual effect due to the presence of the plasma, we compare the results obtained with and without plasma on the full 3D mesh described above. The plasmaless computation is in fact a purely magneto-quasi-static calculation. The number of elements of the mesh is equal to 30907, the number of nodes is 81550. The number of unknowns in the passive structure (i.e., the dimension of the matrix to invert) is 22619. The results are reported in Fig. 3. Evidently, the presence of the plasma has an effect not only on the dynamical properties of the model (e.g., the phase behavior at high frequencies), but also on the static gain (amplitude at zero frequency limit). This is not surprising, since the plasma affects also the magnetostatic coupling between active coils and sensors, because it reacts to external static magnetic field perturbations, so as to reach a different equilibrium configuration and hence, modifying the whole magnetic field map in the surrounding regions.

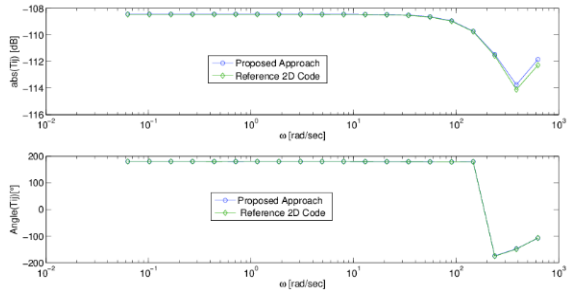

Fig. 2. Comparison of one element of the transfer function $\underline{T}$: proposed approach and reference 2D code.

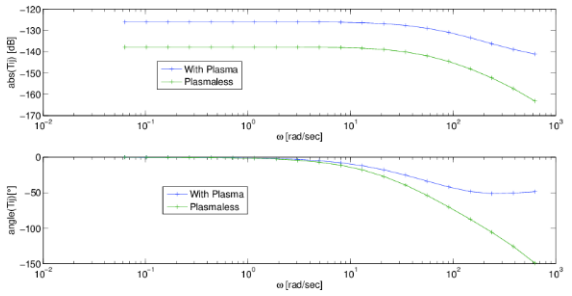

Fig. 3. Effect of plasma on the transfer function.

B. Numerical issues

Regarding the speedup of the matrix assembly, using 25 cores the time required to compute the compressed matrix $\underline{L}$ is about 90 s, the total time required to compute the plasma matrices is about 549 s (540 s of this time is due to the computation of $\underline{Q}$ matrix). In Fig. 4 we report the speedup for assembling $\underline{Q}$ matrix, defined as the assembly time required by one CPU divided by the time obtained using a parallel multi GPUs. Using standard parallel strategy (multi CPUs) the maximum achievable speed up on the proposed computational system is limited to 32.

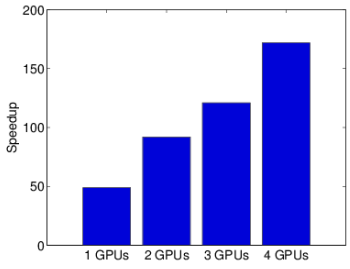

Fig. 4. Speedup for Q-matrix assembly

The time required for each single inversion is about 17.5 s. The total time for all inversions is about 7000s.

The number of iterations required by GMRES to converge increases with the excitation frequency. Without the plasma (i.e., the response due to only the passive structures) the number of iterations required by GMRES to converge is 21 at a frequency of 100 Hz and 9 at 10 Hz. If the plasma is present the number of iteration is 41 at frequency of 100 Hz and 9 at 10 Hz. This is coherent with the general expectation that the used preconditioner is more effective at lower frequencies and without plasma.

V. CONCLUSIONS

We have presented fast parallel techniques for the computation of input-output transfer functions on the RFX-mod fusion devices on hybrid architectures, featuring multiple CPUs and GPUs. The peculiarities of fusion devices make this approach particularly effective in significantly improving the performances of the computation, allowing speed-ups up to almost 200 with respect to standard computations.

REFERENCES

[1] F. Villone, A. G. Chiariello, S. Mastrostefano, A. Pironti, and S. Ventre, "GPU-accelerated analysis of vertical instabilities in ITER including three-dimensional volumetric conducting structures," *Plasma Phys. Control. Fusion*, vol. 54, no. 8, 2012.

[2] G. Rubinacci, S. Ventre, F. Villone, and Y. Liu, "A fast technique applied to the analysis of resistive wall modes with 3D conducting structures," *Journal of Comp. Phys.*, vol. 228, no. 5, pp. 1562-1572, 2009.

[3] P. Sonato, et al., *Fusion Eng. Des*., vol. 66-68, pp. 161, 2003.

[4] F. Villone, et al., "ITER passive and active RWM analysis with the CarMa code," *38th EPS Conference*, paper P5.107, 2011.

[5] R. Fresa, G. Rubinacci, and S. Ventre, "An eddy current integral formulation on parallel computer systems," *Int. Journal for Numerical Methods in Engineering*, vol. 62, no. 9, pp. 1127-1147, 2005.

[6] G. Rubinacci, A. Tamburrino, and S. Ventre, "Fast numerical techniques for electromagnetic non-destructive evaluation," *Nondestr. Testing Eval.*, vol. 24, pp. 165-194, 2009.

[7] R. Albanese and F. Villone, "The linearized CREATE-L plasma response model for the control of current, position and shape in tokamaks," *Nucl. Fusion*, vol. 38, no. 5, pp. 723, 1998.

[8] A. Portone, et al., "Linearly perturbed MHD equilibria and 3D eddy current coupling via the control surface method," *Plasma Phys. Control. Fusion*, 50, 085004, 2008.

[9] A. Capozzoli, et al., "Speeding up aperiodic reflectarray antenna analysis by CUDA dynamic parallelism," *Proc. of the Int. Conf. on Numerical Electromagn. Model. and Opt. for RF, Microwave and Terahertz Appl*., Pavia, Italy, pp. 1-4c, 2014.