Progress In Electromagnetics Research B, Vol. 41, 77–100, 2012

PERFORMANCE ANALYSIS OF PARALLEL NON-ORTHOGONAL PEEC-BASED SOLVER FOR EMC APPLICATIONS

D. Daroui* **and J. Ekman**

Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology, Luleå 97187, Sweden

Abstract—A parallel implementation of a quasi-static Partial Element Equivalent Circuit (PEEC)-based solver that can handle electromagnetic problems with non-orthogonal structures is presented in this paper. The solver has been written in C++ and employs GMM++ and ScaLAPACK computational libraries to make the solver fast, efficient, and adaptable to current parallel computer systems. The parallel PEEC-based solver has been tested and studied on high performance computing clusters and the correctness of the solver has been verified by doing comparisons between results from orthogonal routines and also another type of electromagnetic solver, namely FEKO. Two non-orthogonal numerical test cases have been analysed in the time and frequency domain. The results are given for solution time and memory consumption while bottlenecks are pointed out and discussed. The benchmarks show a good speedup which gets improved as the problem size is increased. With the capability of the presented solver, the non-orthogonal PEEC formulation is a viable tool for modelling geometrically complex problems.

1. INTRODUCTION

Electromagnetic modelling of three-dimensional systems is getting more and more attention due to increasing operational frequency of the modern devices to several gigahertz and electromagnetic interference in complex systems which can cause serious problems. When the operational frequency of electronic devices is increased, magnetic and electrical couplings within and between devices can not be neglected. Electromagnetic compatibility (EMC) standards,

Received 10 April 2012, Accepted 15 May 2012, Scheduled 28 May 2012

* Corresponding author: Danesh Daroui (danesh.daroui@ltu.se).

demand the compliance with limits for electromagnetic interferences. Therefore, studying emission and susceptibility is a vital task in high frequency circuit design.

It has been proven that the Partial Element Equivalent Circuit (PEEC) method [1, 2] is a valid, accurate, and fast solution for various electrical interconnect and packaging (EIP) and EMC problems in the time as well as the frequency domain. Moreover, PEEC-method has been employed to study EMC in various type of problems, including automotive industry [3], power electronics [4] and filter layout optimization [5]. Additionally, extending the PEEC-method to be able to handle non-orthogonal structures by taking advantage of parallel computer systems will provide the possibility to perform EMC analysis on real world problems with complex structures. Further, recent research has extended the basic PEEC-method to handle non-orthogonal cell geometries [6–8] which allows direct modelling of geometrically complex geometries. Unlike for orthogonal structures, partial elements can not be calculated using fast analytical formulations. Therefore, multidimensional integrations are needed to be done numerically for partial inductance and coefficient of potential matrices by maintaining desired accuracy and stability. This shifts the bottleneck in orthogonal PEEC-modelling which is at the solution of the equation system to matrix fill-in for non-orthogonal PEEC-modelling. Many researchers have been working on fast solutions for EM problems, i.e., [9]. For instance, Krylov subspace solver, Fast Multipole Method (FMM) [10] and QR decomposition [11] have been studied in order to accelerate the performance of EM solvers. While different PEEC-based solvers have already been accelerated using other available approaches e.g., FMM [12], wavelet-based solutions [13–15] and QR decomposition, the results usually suffer from restrictions in the stability and low frequency solutions [16]. Alternatively, parallel computer systems, seem to be an ideal solution to speedup both partial element calculation and matrix solution, by dividing the computational load on several high performance computing nodes, without any mentioned restrictions. At the same time, current trends and advances in computer hardware technology support this direction of acceleration techniques. In previous work [17], a parallel PEEC-based solver had been developed which could be used for orthogonal structures. Current work extends the previous solver to be able to analyse more geometrically complex problems. Furthermore, other researchers have successfully implemented parallel electromagnetic solvers [18, 19] in favour of exploiting parallelism to speed up the solution. This paper presents the first parallel implementation of a PEEC-based solver which is appropriate to solve non-orthogonal problems. Section 2

presents the classical PEEC formulations and the extensions to be able to handle non-orthogonal structures. Formulations for partial element calculations are also explained in same section. In Section 3, the process of parallelization of the solver is discussed and libraries and tools which are used to develop the solver is introduced. A numerical technique which have been used to improve the performance of the partial element calculations in non-orthogonal PEEC, is also studied. Section 4 is about validating the correctness of the solver by comparing the results with analytical routines and another electromagnetic simulation tool. In Sections 5 and 6, two simulation test cases are analysed in order to benchmark the performance of the parallel solver. Finally, conclusions and further work is detailed in Section 7.

2. NON-ORTHOGONAL PEEC FORMULATION

This section gives an brief summary of the non-orthogonal PEEC formulation. For further information, see [1, 7, 8, 20, 21].

2.1. Extraction of Equivalent Circuit

The classical PEEC method is derived from the equation for the total electric field at a point [22] written as

$$\mathbf{E}^i(\mathbf{r},t) = \frac{\mathbf{J}(\mathbf{r},t)}{\sigma} + \frac{\partial \mathbf{A}(\mathbf{r},t)}{\partial t} + \nabla\phi(\mathbf{r},t), \tag{1}$$

where $\mathbf{E}^i$ is an incident electric field, $\mathbf{J}$ is a current density, $\mathbf{A}$ is the magnetic vector potential, ϕ is the scalar electric potential, and σ the electrical conductivity all at observation point $\mathbf{r}$.

By using the definitions of the scalar and vector potentials, the current- and charge-densities are discretized by defining pulse basis functions for the conductors and dielectric materials. Pulse functions are also used for the weighting functions resulting in a Galerkin type

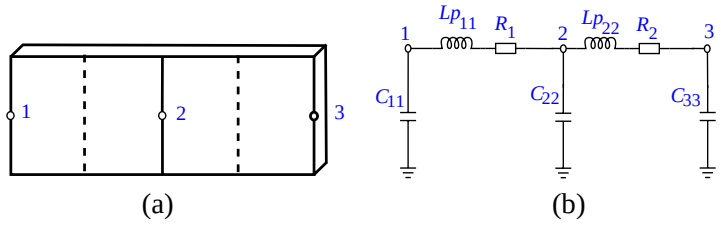


Figure 1. (a) An orthogonal metal strip with 3 nodes and 2 cells, and (b) corresponding PEEC circuit (mutual couplings are not shown).

solution. By defining a suitable inner product, a weighted volume integral over the cells, the field Equation (1) can be interpreted as Kirchhoff's voltage law over a PEEC cell consisting of partial self inductances between the nodes and partial mutual inductances representing the magnetic field coupling in the equivalent circuit. The partial inductances shown as Lp_{11} and Lp_{22} in Figure 1 are defined as

$$\mathbf{L}_{\mathbf{p}_{\alpha\beta}} = \frac{\mu}{4\pi}\frac{1}{a_\alpha a_\beta}\int_{v_\alpha}\int_{v_\beta}\frac{1}{|\mathbf{r}_\alpha - \mathbf{r}_\beta|}dv_\alpha dv_\beta \tag{2}$$

for volume cell α and β. Figure 1 also shows the node capacitances which are related to the coefficients of potential p_{ii}. The coefficients of potentials are computed as

$$\mathbf{P}_{ij} = \frac{1}{S_i S_j}\frac{1}{4\pi\epsilon_0}\int_{S_i}\int_{S_j}\frac{1}{|\mathbf{r}_i - \mathbf{r}_j|}\,dS_j\,dS_i \tag{3}$$

and a resistive term between the nodes, defined as

$$\mathbf{R}_\gamma = \frac{l_\gamma}{a_\gamma \sigma_\gamma}. \tag{4}$$

In (2) and (4), a represents the cross section of the rectangular volume cell normal to the current direction γ, and l is the length in the current direction. Further, v represents the current volume cells and S the charge surface cells.

By assigning a local non-orthogonal coordinate system (a, b, c), general formulations of partial elements will be extended to non-orthogonal form, see [23]. Then as shown in Figure 2 the partial mutual

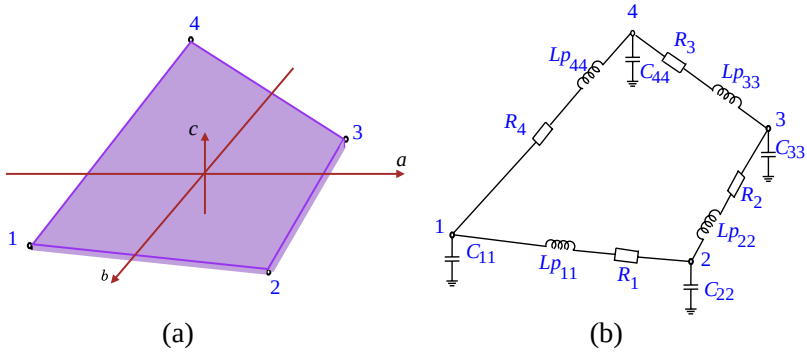


Figure 2. (a) A non-orthogonal metal strip with 4 nodes and 4 cells, and (b) corresponding PEEC circuit (mutual couplings are not shown).

inductances are calculated as

$$\mathbf{L}_{p_{\alpha\beta}} = \mu \int_a \int_b \int_c \int_{a'} \int_{b'} \int_{c'} \hat{a}'.\hat{a} \left| \frac{\partial \mathbf{r_g}}{\partial a} \right| \left| \frac{\partial \mathbf{r_{g'}}}{\partial a'} \right| G(\mathbf{r_g}, \mathbf{r_{g'}}) dv dv'. \quad (5)$$

The coefficient of potential are correspondingly computed as

$$\mathbf{P}_{ij} = \frac{1}{\epsilon} \int_a \int_b \int_{a'} \int_{b'} G(\mathbf{r_g}, \mathbf{r_{g'}}) dA dA'. \quad (6)$$

The double volume integrations in (5) and double surface integration in (6) are carried out coordinates in a hexahedral cell and quadrilateral surface respectively. The free space Green function is used

$$\mathbf{G}(\mathbf{r_g}, \mathbf{r_{g'}}) = \frac{e^{-jk(\mathbf{r_g} - \mathbf{r_{g'}})}}{4\pi |\mathbf{r_g} - \mathbf{r_{g'}}|} \quad (7)$$

which becomes

$$\mathbf{G}(\mathbf{r_g}, \mathbf{r_{g'}}) \approx \frac{1}{4\pi |\mathbf{r_g} - \mathbf{r_{g'}}|} \quad (8)$$

if the time retardation is not included in calculations.

The process of computing partial elements in non-orthogonal models is rigorous and heavy due to the multiple integrations in (5) and (6). Here, a numerical approach using Legendre-Gaussian quadrature is used for the evaluation. In the solver, a variable called Legendre-Gaussian Order (LGO) is defined to be able to choose the order of the weights in the numerical approach. Choosing a higher order will result in more accurate results, but the time taken will increase as well.

2.2. Solution of Equivalent Circuit

The discretization process of the EFIE in (1) and the successive Galerkin's weighting leads to an equivalent circuit formulation. When Kirchhoff's voltage and current laws are enforced to the N_i independent loops and N_ϕ independent nodes of the PEEC equivalent circuit, it is obtained

$$\begin{aligned} &- \mathbf{A}\mathbf{\Phi}(t) - \mathbf{R}\mathbf{i}_L(t) - \mathbf{L}_p \dot{\mathbf{i}}_L(t) = \mathbf{v}_s(t) \\ &\mathbf{P}^{-1}\dot{\mathbf{\Phi}}(t) - \mathbf{A}^T \mathbf{i}_L(t) = \mathbf{i}_s(t), \end{aligned} \quad (9)$$

where

- $\mathbf{\Phi}(t) \in \mathbb{R}^{N_\phi}$ is the vector of node potentials to infinity; $\mathbb{R}^{N_\phi}$ is the node space of the equivalent network;
- $\mathbf{i}_L(t) \in \mathbb{R}^{N_i}$ is the vector of currents including both conduction and displacement currents; $\mathbb{R}^{N_i}$ is the current space of the equivalent network;

- $\mathbf{L}_p$ is the matrix of partial inductances describing the magnetic field coupling;
- $\mathbf{P}$ is the matrix of coefficients of potential describing the electric field coupling;
- $\mathbf{R}$ is the matrix of resistances;
- $\mathbf{A}$ is the connectivity matrix;
- $\mathbf{v}_s(t)$ is the vector of distributed voltage sources due to external electromagnetic fields or lumped voltage sources;
- $\mathbf{i}_s(t)$ is the vector of lumped current sources.

The equation system in (9) is similar to the circuit equations formulated in SPICE-type of solvers for obtaining the solution in node potentials and branch currents. However, for PEECs the equation system in (9) contain more dense matrices ($\mathbf{L}_p$ and $\mathbf{P}$) compared to a pure electric network system solution due to the large number of mutually coupled inductors and capacitances. Therefore, the solution of PEECs requires linear algebra packages suitable for dense matrices. The exception is the full-wave, time domain case where retarded magnetic and electric field couplings are treated as known sources and the $\mathbf{L}_p$ and $\mathbf{P}$ matrices are more sparse [24].

The equation system in (9) is often entitled a Modified Nodal Analysis (MNA) formulation [25] and can be modified to suit the solution of PEECs [24]. From the MNA formulation, the Nodal Analysis (NA) formulation can be derived which only solves for the node potentials by a reduced equation system while the branch currents are calculated in a second step. In the frequency domain the NA system can be written as

$$\mathbf{\Phi} = \left[-\mathbf{A}^T \left(\mathbf{R} + j\omega\mathbf{L}_p\right)^{-1} \mathbf{A} + j\omega\mathbf{P}^{-1}\right]^{-1} \mathbf{I}_S. \tag{10}$$

to solve for the node potentials $\mathbf{\Phi}$ at a specific frequency for the excitation specified by $\mathbf{I}_S$.

2.3. Sequential PEEC-based Solver

A computer software has been developed according to the PEEC method which supports both orthogonal and non-orthogonal models. It is up to the solver to detect whether the model is orthogonal or non-orthogonal and then use proper routines to calculate partial elements. The orthogonality of each surface and volume cell is evaluated, by performing dot-product between the vectors which are along edges at each vertex. If the dot-product of the edge vectors will be less than a small constant e.g., 10^{-9}, then the edges will be orthogonal to each other. Partial elements calculations for non-orthogonal structures are

carried out using numerical routines, while for orthogonal structures analytical routines are invoked [6]. It should be noted that in PEECs, very high accuracy for near and self terms for the partial elements are needed while for far coefficients/couplings the same level of the accuracy is not needed [26]. Therefore a Legendre-Gaussian quadrature will be used with properly chosen order to calculate non-orthogonal partial elements numerically. Higher order will bring better accuracy while the calculation time will be increased.

When the solver is applied, an equivalent circuit is created and

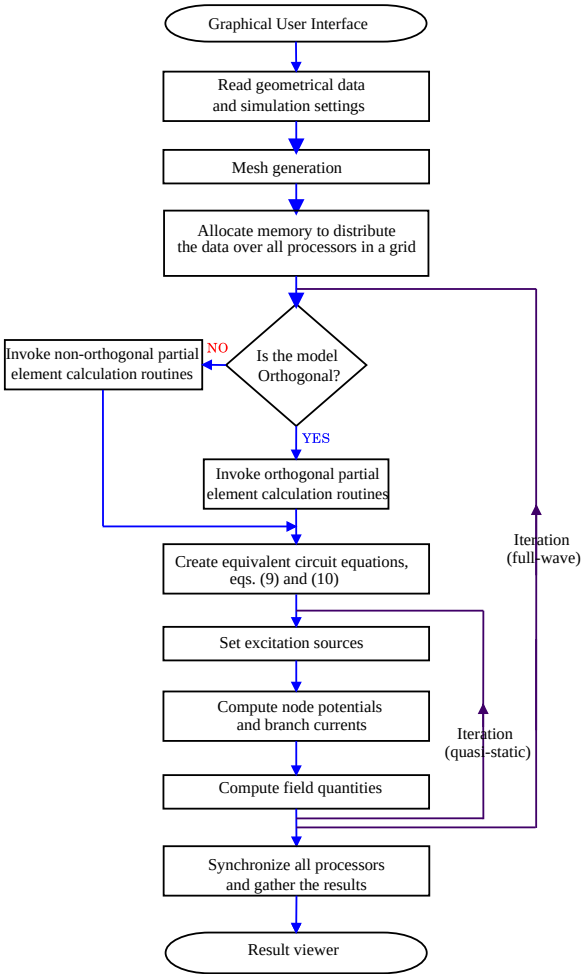


Figure 3. Flow diagram for the parallel PEEC-based solver.

the corresponding resistances, partial inductances, capacitances and coupled voltage and current sources (to account for electromagnetic couplings) for the given geometrical layout (CAD-data as specified in an input file) are calculated. It is also possible to add external electronic (sub-)systems and analysis mode i.e., `.AC` and `.TRAN` as described by the SPICE syntax, to the model. The actual solution of the resulting circuit Equation (9) in either the time- or frequency-domain, is performed in the solver and results are given as current- and potential-distributions in the geometrical layout. Post-processing routines are implemented for calculating field quantities at specified locations. The workflow of the program is shown in Figure 3. The sequential implementation utilizes Intel Math Kernel Library (MKL) package and Intel C++ Compiler in combination with OpenMP directives for multi-core machines. Thus, the program will be able to take advantage of as many processors as available on a multi-processor machine by loop parallelization/vectorization techniques. It is the presented sequential code that has been parallelized and tailored for non-orthogonal PEEC problems, for which results are presented in this paper.

3. PARALLELIZATION OF THE PEEC-BASED SOLVER

This sections discusses the process of parallelizartion of the PEEC-based solver, including computational libraries and methods which were employed.

3.1. Introduction

The main purpose of developing parallel implementation of the PEEC-based solver was to have a scalable solver which can be used to handle very large problems, by having enough resources. Nowadays, real world problems have become very large due to increasing complexity and frequency of electrical systems, which need denser mesh on the structure and thus large number of unknowns to be handled. Hence, the parallel solver becomes more important when such problems are intended to be solved. Besides, when non-orthogonal problems are involved, the solution time increases drastically which in many cases make it infeasible to solve a problem in a reasonable time.

The development platform was a Linux cluster consisting of nodes equipped with two Intel Xeon quad-core 2.5 GHz CPUs and 16 GB of RAM memory. The code has been written in C++ under Linux and is compatible with parallel computer systems with distributed memory architecture using ScaLAPACK [27] as

computational library. ScaLAPACK (Scalable LAPACK) is a library of LAPACK routines, revised for parallel computer systems with distributed memory architecture. The package enables the use of high performance computing clusters in a simple fashion and allows for a considerable acceleration of the developed PEEC-based program. Like LAPACK, ScaLAPACK offers a set of highly optimized routines to solve systems of linear equations, which consists of matrices distributed among a bunch of processors. The library can solve linear matrix equations and perform basic linear algebra operations such as product between matrices and vectors using PBLAS. PBLAS is parallel version of a rich library of computational routines called BLAS which is included in LAPACK. Finally a set of routines called BLACS is used to manage communication between nodes running ScaLAPACK. These routines use algorithms called block-partitioned algorithms to minimize movement of data between nodes by load balancing between computational elements. ScaLAPACK has been written in FORTRAN and developed for parallel computer systems. It is a stable, well tested, and efficient library and provides access to a very large collection of useful, powerful and flexible functions in BLAS and LAPACK which have been parallelized efficiently. Using ScaLAPACK it was assured that a good load balancing is achieved by distributing input data on a number of processing nodes using block cyclic data distribution algorithms [27] which speeds up the operations by minimizing data transfer between processing units.

The parallel solver performs these four steps to solve a problem:

(i) The discretization process is entirely serial and duplicated on all processors.

(ii) The partial element calculations are easily parallelized as no communication is required between nodes while each node calculates assigned part of basic matrices in parallel with other nodes. The main difficulty lies in the mapping between global and local matrix coordinates [19].

(iii) The matrix formulation, (9) or (10), and solution parts are implemented using ScaLAPACK routines.

(iv) At the end when all processes has reached final synchronization point, the results will be gathered on the root processing unit and will be saved in appropriate format.

3.2. Parallelization of Partial Element Computations

The partial element calculations are easily parallelized using parallel processors which fill a large matrix, distributed by ScaLAPACK data management algorithms, completely in parallel and independent of

each other. For the time domain problems these are $\mathbf{L}_p$ and $\mathbf{P}$ matrices which are symmetric and have entries of type double precision floating-point. Hence the fill-in times for the time domain solver is decreased linearly as number of allocated processors grows. Since these two matrices are symmetric, Cholesky factorization [28] routines in ScaLAPACK package can be used to factorize them.

Due to the properties of the entries of the $\mathbf{L}_p$ and $\mathbf{P}$ matrices which has the complex data type for problems in frequency domain and because these matrices do not fulfil Hermitian properties, Cholesky factorization can not be applied. Thus, only LU factorization is possible as a direct strategy to solve the equation and therefore the whole matrix needs to be filled. The process of placing element from one part of a distributed matrix to the other part is computationally expensive and complicated in parallel programs and especially for the MNA-approach seen in (9). But this was overcome by a special Transpose-and-Add algorithm as detailed in [29].

Although partial element calculations are quite fast for orthogonal models, it usually takes much longer time for non-orthogonal structures due to multiple folded integrals in the Equations (5) and (6). The time taken to calculate these values, are highly dependent to the value of LGO, as stated in Section 2.1. Table 1 lists the time taken to calculate partial elements, namely $\mathbf{L}_p$ and $\mathbf{P}$ matrices, for a $100 \times 100 \times 0.1\,\mathrm{cm}$ rectangular plate with different mesh, when orthogonal routine and non-orthogonal routines are used. In this test, the LGO is set to 3 when non-orthogonal routines are used. Although, a small LGO is used in this test, but the difference between taken time when different types of routines are used is already substantial.

Table 1. Time taken to calculate partial elements for an orthogonal and non-orthogonal structure.

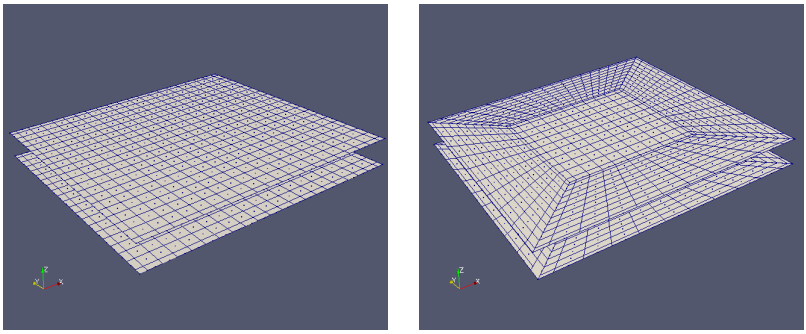
Model	Num. of cells for volume	Num. of cells for surface	Time [s] L_p	Time [s] P
Orthogonal	2 121	1 050	9	2
Non-Ortho.	2 121	1 050	64	19
Orthogonal	4 681	2 170	29	9
Non-Ortho.	4 681	2 170	313	72
Orthogonal	6 336	2 880	46	14
Non-Ortho.	6 336	2 880	566	150

3.3. Parallelization of Matrix Solutions

After filling-in the matrices, the solution of the time- or frequency-domain versions of the circuit equations in (9) and (10) has to be performed. This is done using the ScaLAPACK library of high performance linear algebra routines for distributed-memory message-passing MIMD (multiple instruction stream, multiple data stream) computers and networks of workstations supporting parallel virtual machine (PVM) and/or message passing interface (MPI). ScaLAPACK uses block cyclic data distribution [30] to achieving good load balancing. This means that matrices are divided into blocks in two dimensions and these blocks are assigned to a set of processors. This is further detailed in [19] when using the numerical electromagnetics code (NEC) to solve electromagnetic problems using ScaLAPACK.

4. VALIDATION OF NON-ORTHOGONAL FORMULATION

The presented non-orthogonal PEEC-based solver is an extension of a previously developed orthogonal PEEC-based solver that has been verified, for correctness, by comparison with different measurements [31, 32] and analytical calculations. In order to verify the correctness of the non-orthogonal routines for partial elements calculations, two $100\times100\times0.1$ cm square plates, placed parallel to each other at a distance of 10 cm, with different meshing (both orthogonal and non-orthogonal) are used. Figure 4 depicts the tested plates with orthogonal and non-orthogonal mesh applied on it. In former case, well



(a) (b)

Figure 4. (a) Parallel plates with orthogonal mesh, and (b) same plate with non-orthogonal mesh.

tested orthogonal routines for partial element calculations are used, while in later case non-orthogonal routines are called. The plates have been modelled to be very thin which means that it has been assumed that there is no current along thickness of the plates. In this test, a voltage source connects two plates at the near end corner and a 50 Ω resistor connects the plates at the far end corner. The simulation has been performed in the frequency range of 1 to 100 MHz. Since the orthogonal routines has already been verified with the measurement, so a good agreement between results from orthogonal and non-orthogonal routines, will also ensure the correctness of the non-orthogonal PEEC-based solver.

Figures 5 and 6 demonstrate the simulation results for real and imaginary parts of the voltage of the resistor which connects two parallel plates at the far end, when the LGO is equal to 3 for non-orthogonal functions. The plots show the results using orthogonal and non-orthogonal engines of the PEEC-based solver and also results from FEKO [33], a commercial electromagnetic simulation tool. It can be observed that there is a good agreement between the results for the orthogonal and non-orthogonal structures, as well as results from the FEKO solver. Since FEKO uses Method of Moments-approach (MoM) [34], only a surface mesh is applied, while PEEC uses both a surface and a volume mesh which can explain the differences in the results. On the other hand, the small error in non-orthogonal results, is due to the numerical approach which is used to reduce the computation time for non-orthogonal formulations, while orthogonal routines use analytical approaches. Using PEEC-based solver the

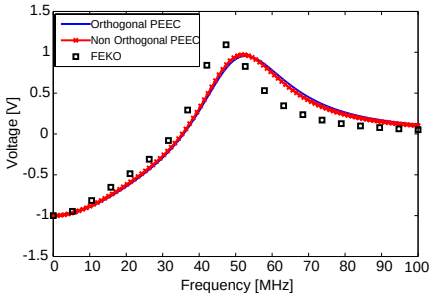


Figure 5. Real part of the voltage of the resistor between two plates, using orthogonal and non-orthogonal PEEC and FEKO.

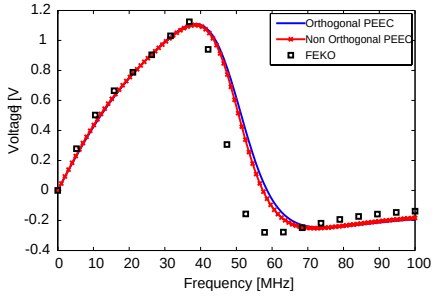


Figure 6. Imaginary part of the voltage of the resistor between two plates, using orthogonal and non-orthogonal PEEC and FEKO.

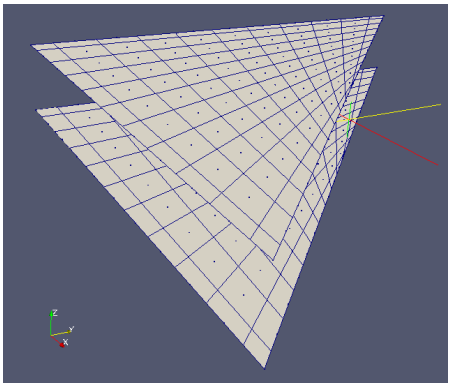


Figure 7. The non-orthogonal parallel triangular plates modelled in PEEC.

problems consisted of 683 and 579 unknowns for orthogonal and non-orthogonal cases respectively, whereas, number of unknowns in the simulated model in FEKO was 464.

5. NUMERICAL TEST (I) — TRIANGULAR PLATES

Previously, the speedup factor of the parallel PEEC-based solver for orthogonal structures have been discussed in [17]. Here, in the first numerical test, the new parallel non-orthogonal PEEC-based solver is tested with a simple non-orthogonal model which consists of two parallel triangular plates which are shown in Figure 7. The dimension of each plate is $2 \times 2 \times 2.8$ cm with the thickness of 0.1 µm. The distance between the plates is 0.3 cm and it has been assumed that there will be no current along thickness of the plates. The plates are excited using a voltage source at the near end corner and connected through a 50 Ω resister at the far end corner.

From the calculated matrices, as briefly detailed in Section 2.2, two popular methods are used to formulate the circuit equations for the PEEC model. First, the MNA formulation as shown in (9) and second the NA formulation as shown in (10). The MNA formulation is the more general of the two and preferred mainly due to its ability to handle general circuit element inclusion with the PEEC model [25] and a stable low frequency behaviour. This test case has been simulated using the quasi-static MNA solver and the simulations have been run, up to 300 MHz in the frequency domain with 10 samples and up to 5 ns in the time domain using 400 time steps.

Table 2. Characteristics of the triangular plates.

Test	Number of volume cells (N_i)	surface cells (eq. nodes) (N_ϕ)	unknowns ($N_i + N_\phi$)
t_1	1 680	882	2 562
t_2	3 720	1 922	5 642
t_3	6 560	3 362	9 922
t_4	25 920	13 122	39 042

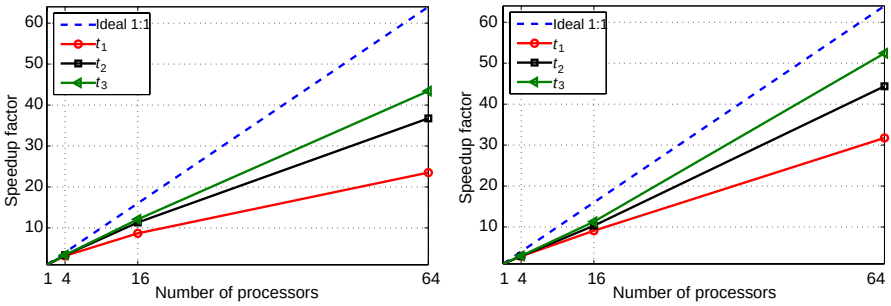


Figure 8. Speedup factor for partial element calculation and fill-in for (a) LGO = 1, and (b) LGO = 6.

5.1. Partial Element Calculation Speedup

The speedup analysis is based on four different test cases with different meshes and number of unknowns for the triangular plates. The characteristics of each tested model are shown in Table 2 where number of nodes are equal to number of surface cells since simulated structures are assumed to have zero thickness.

The simulations have been done using different LGO as stated in Section 2.1. The partial element calculations are efficiently parallelized as seen in Table 3 for LGO = 1 and Table 4 for LGO = 6. The performance gain for the parallel implementation, as shown in Tables 3 and 4, can be displayed by using a speedup factor

$$S(n) = \frac{t_{p_1}}{t_{p_n}}. \tag{11}$$

where t_{p_1} is the time taken by the parallel code using one processor and t_{p_n} is the time taken by the parallel code using n processors. These are shown in Figure 8 and compared with the exact linear speedup

Table 3. Partial element calculation times [s] for LGO = 1.

Number of processors	t_1	t_2	t_3	t_4
1	55	283	955	−*
4	17	85	282	1 093
16	6	25	80	286
64	2	8	22	87

* *Not available due to memory limitations*

Table 4. Partial element calculation times [s] for LGO = 6.

Number of processors	t_1	t_2	t_3	t_4
1	455	2 218	6 973	−*
4	159	764	2 360	8 581
16	50	215	619	2 181
64	14	50	133	594

* *Not available due to memory limitations*

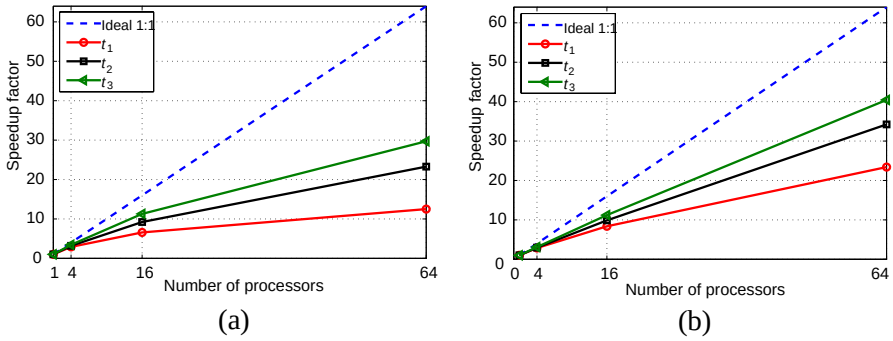


Figure 9. Total PEEC-model solution time for frequency domain simulations for (a) LGO = 1 and for (b) LGO = 6.

(ideal 1:1), which scales linearly as number of processors increases, in an ideal situation.

From the figures it is clear that the fill-in time has been parallelized in a more efficient way when higher LGO is used. The reason is that as

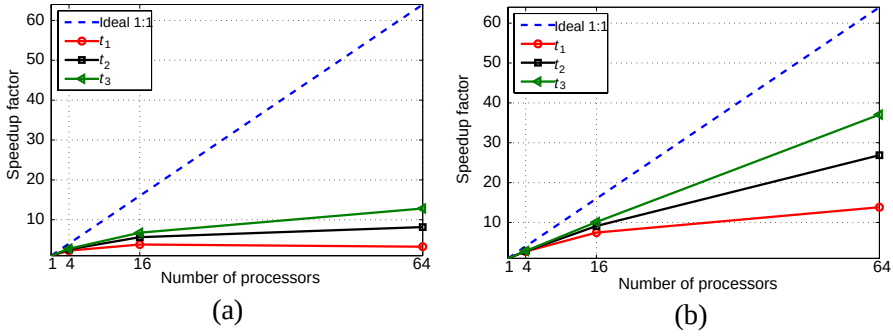


Figure 10. Total PEEC-model solution time for frequency domain simulations for (a) LGO = 1 and for (b) LGO = 6.

Table 5. Time for the total solution for time domain simulations.

Number of processors	Time [s] t_1 LGO = 1	t_1 LGO = 6	t_2 LGO = 1	t_2 LGO = 6	t_3 LGO = 1	t_3 LGO = 6	t_4 LGO = 1	t_4 LGO = 6
1	68	470	343	2 285	1 127	7 151	−*	−*
4	30	172	134	811	413	2 498	1 838	9 331
16	18	63	61	251	167	707	6 09	2 501
64	21	34	42	85	88	193	300	807

* *Not available due to memory limitations*

LGO increases, more calculations are needed for compute each value. Moreover, the time spent for computations will dominate over the communication overhead and latency between processing nodes which results in better speedup.

5.2. Total PEEC-model Solution Time

To analyse the overall performance of the parallel solver, solution time for all test cases are shown in Tables 5 and 6. Then, the speedup factors for total PEEC-model solutions are given. This is shown in Figure 9 for the frequency domain and in Figure 10 for the time domain, both implementations of the MNA method.

According to the results depicted in Figure 9, the speedup factor has been linearly improved, as number of processors grows. Table 6 even shows a superlinear speedup for example for the first test case with LGO set to 1, when the number of processors is ×4, but the speedup

Table 6. Time for the total solution for frequency domain simulations.

Number of processors	Time [s]							
	t_1		t_2		t_3		t_4	
	LGO = 1	LGO = 6	LGO = 1	LGO = 6	LGO = 1	LGO = 6	LGO = 1	LGO = 6
1	125	524	694	2 908	4 633	10 636	−*	−*
4	25	25	308	987	1 248	3 429	12 403	19 815
16	19	12	80	295	411	949	3 465	5 375
64	10	7	42	85	156	263	1 120	976

* *Not available due to memory limitations*

factor is 5 which is more than ideal speedup. Although, superlinearity happens rarely, but this is usually due to maximum cache hit and best memory alignment for some problems, when the data is perfectly aligned in the memory which minimizes the fetch time as well as cache loss [35].

Despite higher computational complexity i.e., $O(n^3)$ of matrix solving, comparing to matrix fill-in i.e., $O(n^2)$, filling matrices are the most time consuming part in the presented study. This is due to the multiple nested loops to approximate the folded integrals in the non-orthogonal formulations, i.e., (5) and (6) and also expensive computations which need to be performed to calculate each element. Additionally, the time complexity for calculating each element will also be increased from around $O(16)$ for orthogonal solutions to approximately $O(18L)$ for non-orthogonal, where L is equal to the LGO parameter.

Comparison between Tables 3, 4, 5, and 6 also reveals that unlike orthogonal solver where the most of the solution time was consumed in the solution of the equations system [17], in non-orthogonal solver, the bottleneck lies in the partial element calculations part. This conclusion is confirmed by observing that changing LGO to higher levels has a high impact on the total solution time while changing LGO will affect only partial element calculation process and not the matrix solution. This is completely natural, since fast analytical solutions are not applicable for partial element calculations when non-orthogonal models are being analysed and hence cumbersome integrals are needed to be calculated numerically. Therefore, for non-orthogonal solutions, assigning more processors, even if the required memory could be achieved using less nodes, would decrease the solution time. This is not necessarily true for orthogonal problems since the speedup property of the solution gets

saturated almost quickly when number of processors grows because of the fast partial element calculation.

In addition, results show better improvement for larger problems as well as higher level of LGO while smallest problem in Figure 9(a) shows a saturation which means that maximum level of parallelism has already been achieved with the allocated resources. This means using more processing units to solve such a small problem will not necessarily improve the performance and in worst case can even degrade the solution time.

Obviously, results from the time domain solver are quite different. For example Figure 10(a) shows that almost all solutions are already saturated. It can be observed that the solution time for the smallest problem is even degraded when number of processors has been increased. This can be explained, because of relatively less expensive calculations in time domain, specially with small LGO, the communication time will dominate. Despite the test with the small LGO, which did not show any proper parallelization, but with higher LGO, as depicted in Figure 10(b), good speedup is gained.

From the figures, several conclusions can be drawn:

- For small problems in time domain, increasing the number of processors did not improve the overall solution time, since the communication time between the processors increases and exceed the total solution time. For example in Figure 10(a), problems t_1 and t_2 have been saturated at 16 processors. Therefore, using multiple processors for a small problem will not necessarily improve the performance.
- According to Table 6, the frequency domain problems experience a larger speedup factor compared to time domain problem. However, in general, frequency domain problems are more time consuming in absolute numbers.
- Legendre-Gaussian Order (LGO) plays an important role in speedup factor. Using larger LGO, however will increase the accuracy of the solution and also the solution time, but better speedup is gained. Choosing a proper LGO is usually a trade-off between accuracy and computational resources. According to experiments, in most cases, when LGO is set to 3, the results are accurate enough.

6. NUMERICAL TEST (II) — SPHERE

In the last test case, a sphere with the diameter of 2 cm, shown in Figure 11, is studied. Capacitance calculations are of great importance

in many application areas, for example in the analysis of electrical interconnects and packaging (EIP) [36]. To some extent analytical routines can be used, but for more complex geometries, numerical techniques are a valuable tool. Before any analysis regarding the performance of the solver, the correctness of the simulation results are verified. This verification is done by comparing the capacitance of the sphere, using analytical formula and compare it with the results from the PEEC-based solver. The capacitance of a sphere with the radius of R is calculated as

$$C = 4\pi\epsilon_0 R \tag{12}$$

where ϵ_0 is the vacuum permittivity and R is the radius of the sphere. Using (12), the capacitance for the studied sphere is calculated as 1.11 pF. Next, the non-orthogonal PEEC-solver is used to calculate the capacitance of the sphere. Figure 12 depicts the comparison between analytical and PEEC results. The mesh level on the x-axis represents how the structure is disctritized into cells, so the higher level of the mesh, the finer mesh is used and the model is divided into smaller parts to increase the accuracy. The LGO has been assumed to be 3 in this test. It can be observed that when applied mesh on the sphere is coarse, the results from the PEEC-based solver are quite inaccurate. As the applied mesh gets finer, the results get closer to the correct value which end up with very good agreement between two approaches.

Table 7 shows characteristics of the sphere test cases. The main

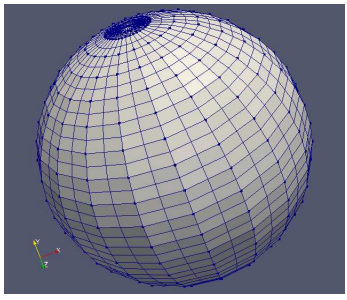


Figure 11. The sphere modelled in PEEC.

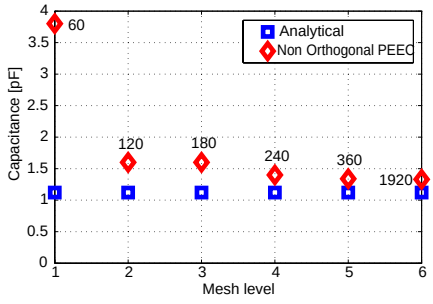


Figure 12. Comparison between capacitance of a sphere with 2 cm diameter using analytical values and numerical results from PEEC with different mesh. The number of surface cells is stated close to each simulation point.

Table 7. Sphere test characteristics.

Test	Number of volume cells (N_i)	nodes (N_ϕ)	unknowns ($N_i + N_\phi$)	Memory [GB] TD	FD
S_1	25 920	10 362	36 282	27	35
S_2	50 400	19 562	69 962	114	184
S_3	85 827	33 618	119 445	308	475

Table 8. Time for the total solution.

Test	TD [s]	FD [s]	Number of processors
S_1	1 040	4 208	64
S_2	1 612	6 750	240
S_3	3 339	20 210	440

aim of these simulations was to stress the code as much as possible by solving very large problems with hundred thousands of unknowns. The memory consumption of each test gives an idea about the size of the problem. It can also be concluded that time domain solver consumes less memory due to the complex data type which is used for frequency domain solver and needs more memory to store both real and imaginary part of the numbers. In these tests, the sphere is excited at the top using a current source and a $50\,\Omega$ resistor is connected at the bottom. The source and the resistor are also grounded. The simulations has been carried out using Nodal Analysis (NA) solver and in both time- and frequency domain, within 10 frequency and 400 time steps.

Table 8 demonstrates the total solution time for each test in each domain and how many processors have been used to solve the problem. It is evident that the largest problem has been solved in a reasonable time with 440 processors and 475 GB memory. It also can be observed that the time taken by the time domain solver is substantially lower than the frequency domain solver. Same conclusion can be made about the memory usage for each solver.

7. CONCLUSIONS AND FURTHER WORK

This paper presented the first implementation of a parallel non-orthogonal PEEC-based solver and the performance of the solver, with respect to the computational aspects has been studied. It was

demonstrated that the new solver can handle very large problems with non-orthogonal structure with about 120000 unknowns. The developed solver utilized GMM++ and ScaLAPACK packages as computational libraries on high performance computing clusters with distributed memory architecture.

Two numerical test cases were analysed in order to study the performance of the solver on parallel computer systems. The results proved that:

- Problems in the frequency domain need more memory as well time, comparing to time domain problems and they are computationally heavier to be solver.
- In order to solve small problems, it is usually more efficient to allocate small number of processors. Using more processors than needed for relatively small problems, can degrade the overall solution because the communication overhead between processors and the latency will dominate.
- Frequency domain solver experienced better speedup than time domain solver.
- Unlike orthogonal models, non-orthogonal models shifted the bottleneck of the solution from system equation solution, to partial element calculation, since partial elements in non-orthogonal structures need cumbersome and numerical formulations to be calculated.

In next step of the performance acceleration, using Krylov subspace methods e.g., iterative solvers with proper preconditioning methods can be considered. Due to lower time complexity of iterative solvers i.e., $O(n^2)$, over direct solvers i.e., $O(n^3)$, these methods are attractive to be used to solve very large problems. Additionally, taking advantage of multi-core processors on shared memory systems, will make it possible to exploit parallelism on personal computers and hence speed up the solution process. Smart sparsification techniques can also be applied to reduce memory usage without loosing accuracy and still having stable results.

REFERENCES

1. Ruehli, A. E., "Equivalent circuit models for three dimensional multiconductor systems", *IEEE Trans. on Microwave Theory and Techniques*, Vol. 22, No. 3, 216–221, The original PEEC (partial element equivalent circuit) paper, Mar. 1974.

2. Song, Z. F., D. L. Su, F. Duval, and A. Lous, "Model order reduction for PEEC modeling based on moment matching," *Progress In Electromagnetics Research*, Vol. 114, 285–299, 2011.
3. Frank, F., M. Zitzmann, G. Steinmair, and R. Weigel, "Methods for circuit-based automotive EMC simulation incorporating VHDL-AMS models," *Proc. of Asia-Pacific Symp. on EMC*, Singapore, 2008.
4. Thamm, S., S. V. Kochetov, G. Wollenberg, and M. Leone, "PEEC modeling for EMC-relevent simulations of power electronics," *Proc. of 17th Int. Conf. on Radioelektronika*, Brno, Czech Republic, 2007.
5. De Oliveira, T., J. Guichon, J. Schanen, and L. Gerbaud, "PEEC-models for EMC layout optimization," *Proc. of 6th Int. Conf. on Integrated Power Electronics Systems (CIPS)*, Nüremberg, Germany, 2010.
6. Müsing, A., J. Ekman, and J. W. Kolar, "Efficient calculation of non-orthogonal partial elements for the PEEC method," *IEEE Trans. on Magnetics*, Vol. 45, No. 3, 1140–1143, Mar. 2009.
7. Antonini, G., A. Orlandi, and A. E. Ruehli, "Analytical integration of quasi-static potential integrals on nonorthogonal coplanar quadrilaterals for the PEEC method," *IEEE Trans. on Electromagnetic Compatibility*, Vol. 44, No. 2, 399–403, 2002.
8. Antonini, G., A. Ruehli, and J. Esch, "Non orthogonal PEEC formulation for time and frequency domain modeling," *Proc. of the IEEE Int. Symp. on Electromagnetic Compatibility*, Minneapolis, MN, Aug. 2002.
9. Chew, W. C., J. M. Jin, C. C. Lu, E. Michielssen, J. M. Song, "Fast solution methods in electromagnetics," *IEEE Trans. on Antennas and Propagation*, Vol. 45, No. 3, 533–543, 1997.
10. Engheta, N., W. D. Murphy, V. Rokhlin, and M. S. Vassilou, "The fast multipole method (FMM) for electromagnetic scattering problems," *IEEE Trans. on Antennas and Propagation*, Vol. 40, No. 6, 634–641, Jun. 1992.
11. Kapur, S. and D. Long, "IES^3: A fast integral equation equation solver for efficient 3-dimensional extraction," *Int. Conf. on Computer Aided Design*, 448–455, Nov. 1997,
12. Antonini, G., "Fast multipole formulation for PEEC frequency domain modeling," *Journal Applied Computat. Electromag. Society*, Vol. 17, No. 3, Nov. 2002.
13. Antonini, G., A. Orlandi, and A. Ruehli, "Speed-up of PEEC method by using wavelet transform," *Proc. of the IEEE Int. Symp.*

on Electromagnetic Compatibility, Washington, DC, Aug. 2000.

14. Antonini, G., A. Orlandi, and A. E. Ruehli, "Fast iterative solution for the wavelet-PEEC method," *Proc. of the International Zurich Symposium on Electromagnetic Compatibility*, Zürich, SW, Feb. 2001.
15. Antonini, G. and A. Orlandi, "Computational properties of wavelet based PEEC analysis in time domain," *Proc. of Applied Computational Electromagnetics Society Conf.*, Monterey (CA), USA, Mar. 2000.
16. Antonini, G. and A. E. Ruehli, "Fast multipole method and multifunction PEEC methods," *IEEE Trans. on Mobile Computing*, Vol. 2, No. 4, 288–298, Dec. 2003.
17. Daroui, D. and J. Ekman, "Parallel implementation of the PEEC method, *Journal Applied Computat. Electromag. Society*, Vol. 25, No. 5, 410–422, 2010.
18. Hanawa, T., M. Kurosawa, and S. Ikuno, "Investigation on 3-D implicit FDTD method for parallel processing," *IEEE Trans. on Magnetics*, Vol. 41, No. 5, 1696–1699, May 2005.
19. Rubinstein, A., F. Rachidi, M. Rubinstein, and B. Reusser, "A parallel implementation of NEC for the analysis of large structures," *IEEE Trans. on Electromagnetic Compatibility*, Vol. 45, No. 2, May 2003.
20. Ruehli, A. E., "Inductance calculations in a complex integrated circuit environment," *IBM Journal of Research and Development*, Vol. 16, No. 5, 470–481, Sep. 1972.
21. Ruehli, A. E. and P. A. Brennan, "Efficient capacitance calculations for three-dimensional multiconductor systems," *IEEE Trans. on Microwave Theory and Techniques*, Vol. 21, No. 2, 76–82, Feb. 1973.
22. Ramo, S., J. R. Whinnery, and T. Van Duzer, *Fields and Waves in Communication Electronics*, WILEY, 1994.
23. Ruehli, A. E., G. Antonini, J. Esch, J. Ekman, A. Mayo, and A. Orlandi, "Non-orthogonal PEEC formulation for time and frequency domain EM and circuit modeling," *IEEE Trans. on Electromagnetic Compatibility*, Vol. 45, No. 2, 167–176, May 2003.
24. Antonini, G., J. Ekman, and A. Orlandi, "Full wave time domain PEEC formulation using a modified nodal analysis approach," *Proc. of EMC Europe*, Eindhoven, The Netherlands, 2004.
25. Ho, C., A. Ruehli, and P. Brennan, "The modified nodal approach to network analysis," *IEEE Trans. on Circuits and Systems*, 504–509, The original MNA paper, Jun. 1975.

26. Antonini, G., J. Ekman, A. Ciccomancini Scogna, A. E. Ruehli, "A comparative study of PEEC circuit elements computation," *Proc. of the IEEE Int. Symp. on EMC*, Istanbul, Turkey, 2003.
27. Choi, J., J. J. Dongarra, R. Pozo, and D. Walker, "ScaLAPACK: A scalable linear algebra library for distributed memory concurrent computers," *Proc. of the Fourth Symp. on the Frontiers of Massively Parallel Computation*, IEEE Computer Society Press, 1992.
28. Gratton, S., "Graphics card computing for cosmology: Cholesky factorization," *Proc. of IEEE 10th Conf. on Computer and Information Technology*, Bradford, UK, 2010.
29. Daroui, D., "Performance of integral equation based electromagnetic analysis software on parallel computer systems," MS thesis, University of Gothenburg, Feb. 2007.
30. Dongarra, J. J. and D. W. Walker, "The design of linear algebra libraries for high performance computers," No. ORNL/TM-12404, University of Tennessee, Knoxville, TN, USA, 1993, citeseer.ist.psu.edu/article/dongarra93design.html.
31. Daroui, D., I. Stevanović, D. Cottet, and J. Ekman, "Bus bar simulations using the PEEC method," *Proc. of 26th Int. Review of Progress in Applied Computational Electromagnetics ACES*, Tampere, Finland, 2010.
32. Cottet, D., I. Stevanović, B. Wunsch, D. Daroui, J. Ekman, and G. Anotinini, "EM simulation of planar bus bars in multi-level power converters," *Proc. of EMC Europe*, Rome, Italy, 2012.
33. FEKO-Electromagnetic simulation software, Available Online: http://www.feko.info.
34. Harrington, R. F., *Time-Harmonic Electromagnetic Fields*, the method of moments reference, McGraw-Hill Book Co., 1961; New Edition, Krieger, 1982.
35. Helmbold, D. P. and C. E. McDowell, "Modeling speedup (n) greater than n," *IEEE Trans. on Parallel and Distributed Systems*, Vol. 1, No. 2, 250–256, Apr. 1990.
36. Avinash, S., B. N. Joshi, and A. M. Mahajan, "Analysis of capacitance across interconnects of low-K dielectric used in a deep sub-micron CMOS technology," *Progress In Electromagnetics Research Letters*, Vol. 1, 189–196, 2008.